VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV INFORMAČNÍCH SYSTÉMŮ
DEPARTMENT OF INFORMATION SYSTEMS

KOMPLEXNÍ ON-LINE TRÉNINKOVÝ DENÍK

COMPLEX ON-LINE TRAINING DIARY

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — **Bc. ZDENĚK KAMENSKÝ**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. VLADIMÍR BARTÍK, Ph.D.**
SUPERVISOR

BRNO 2021

Vysoké učení technické v Brně
Fakulta informačních technologií

Ústav informačních systémů (UIFS) Akademický rok 2020/2021

Zadání diplomové práce

Student: **Kamenský Zdeněk, Bc.**
Program: Informační technologie
Obor: Informační systémy
Název: **Komplexní on-line tréninkový deník**
Complex On-Line Training Diary
Kategorie: Informační systémy

Zadání:

1. Seznamte se s tvorbou webových aplikací, aplikací pro chytré hodinky a možnostmi komunikace webových aplikací s chytrými hodinkami různých značek. Dále prostudujte problematiku získávání znalostí z dat.
2. Prostudujte existující řešení v oblasti aplikací zaznamenávající tréninky, tréninkové plány atd.
3. Navrhněte aplikaci pro záznam tréninků chytrými hodinkami, které kromě toho bude mít rozhraní pro trenéry a různé tréninkové skupiny a bude umožňovat analýzu dat ze záznamů tréninků.
4. Navrženou aplikaci implementujte a otestujte její funkčnost.
5. Zhodnoťte dosažené výsledky a další možnosti pokračování tohoto projektu.

Literatura:

- Han, J., Kamber, M.: Data Mining: Concepts and Techniques. Third Edition. Morgan Kaufmann Publishers, 2012, 703 p., ISBN 978-0-12-381479-1.
- Daniel, S.F.: Android Wearable Programming. Packt Publishing, 2015. ISBN 978-1-78-528015-3.

Při obhajobě semestrální části projektu je požadováno:

- Body 1 až 3.

Podrobné závazné pokyny pro vypracování práce viz https://www.fit.vut.cz/study/theses/

Vedoucí práce: **Bartík Vladimír, Ing., Ph.D.**
Vedoucí ústavu: Kolář Dušan, doc. Dr. Ing.
Datum zadání: 1. listopadu 2020
Datum odevzdání: 19. května 2021
Datum schválení: 27. října 2020

Abstrakt

Cílem této diplomové práce bylo navrhnout a posléze vytvořit on-line tréninkový deník pro sportovce. Nejprve proběhlo ujasnění některých klíčových pojmů týkajících se daného tématu včetně celé oblasti dolování dat a jeho využití. Poté bylo nutné zanalyzovat stávající řešení na poli sportovních aplikací, zvážit požadavky potenciálních uživatelů a v neposlední řadě celý systém navrhnout, implementovat a otestovat. Pro analýzu sportovních dat a záznamů aktivit určenou jak pro jednotlivé atlety, tak pro jejich trenéry na úrovni týmů byly použity nejrůznější metody dolování dat.

Abstract

Design and implementation of online training diary for athletes is the main goal of this thesis. At the beginning, it was necessary to explain some of key words, related to the thesis topic. One of the most important things is data mining and its usability for sports data analysis. After that, existing solutions of sport applications were analyzed and also it was obligatory to analyze potential users requirements. Application design, implementation and testing were the next steps. Some of data mining methods were used for analysis of sports data intended for individual athletes and their coaches.

Klíčová slova

Web, informační systém, sport, sportovní aplikace, tréninkový deník, databáze, dolování dat, analýza dat

Keywords

Web, information system, sport, sport application, training diary, database, data mining, data analysis

Citace

KAMENSKÝ, Zdeněk. *Komplexní on-line tréninkový deník*. Brno, 2021. Diplomová práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Vladimír Bartík, Ph.D.

Komplexní on-line tréninkový deník

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením pana Ing. Vladimíra Bartíka, Ph.D. Uvedl jsem všechny literární prameny, publikace a další zdroje, ze kterých jsem čerpal.

.....
Zdeněk Kamenský
22. dubna 2021

Poděkování

Rád bych poděkoval vedoucímu práce panu Ing. Vladimíru Bartíkovi, Ph.D. za vedení při tvorbě této diplomové práce, dále pak své přítelkyni a rodině za podporu.

Obsah

Kapitola 1

Úvod

Cílem této diplomové práce je vytvořit komplexní webovou aplikaci či webový informační systém, fungující jako on-line tréninkový deník nebo zápisník, určený pro sportovce všech výkonnostních skupin (dále jen aplikace, systém nebo informační systém). Využití nalezne jak u atletů, tak u výkonnostních a volnočasových sportovců. Významnou roli může hrát zejména u organizovaných týmů, oddílů a tréninkových skupin, kde sportovci trénují pod vedením trenéra. Ten, díky této aplikaci, má možnost vidět nejrůznější statistiky výkonu, odtrénovaných objemů, intenzit, srdečního tepu a dalších zajímavých ukazatelů svých svěřenců. Z těchto dat pak může snadněji připravovat další tréninky a tréninkové plány šité na míru jednotlivým atletům. Pro samotné sportovce je to také velmi užitečné. Mohou přehledně vidět, jak se jejich výkonnost v lepším případě zvyšuje, nebo v tom horším snižuje. Zjistí, jaké tréninkové metody jim vyhovují a které jsou naopak neefektivní. Přidanou hodnotou je zcela jistě i to, že sportovec vidí, jak si vede na podobně náročných trasách a jak je na tom v porovnání s ostatními členy týmu. Aplikace je aktuálně primárně zaměřena na všechny typy běhu, nicméně použití pro jiné sporty, jako cyklistiku, chůzi, turistiku, lezení a další, je také možné.

V prvé řadě je nutné zajistit import sportovních záznamů uživatele do aplikace. V dnešní době již drtivá většina atletů, výkonnostních sportovců nebo jen lidí, kteří se rádi aktivně hýbou, využívá nejrůznější moderní technologie k zaznamenávání sportovních dat. Člověk se jde proběhnout do parku se psem a vše zaznamená do chytrého telefonu, chytrých hodinek, nebo přímo k tomu určeného sofistikovaného sporttesteru. Po skončení aktivity takový uživatel s oblibou sleduje, jak se mu dařilo, jakým tempem běžel, jak moc byla trasa do kopce, nebo naopak z kopce a tato data třeba sdílí s přáteli na sociálních sítích. V této práci bylo tedy klíčové zjistit, jaké technologie sportovci používají nejčastěji, aby využití aplikace nebylo minoritní záležitostí. Dále bylo nutné objevit, jakým způsobem je možné z aplikací a zařízení pro záznam aktivit extrahovat data o sportovní aktitivě, jaký formát je k tomu vhodný, aby nedocházelo k zániku důležitých dat, jako je například vývoj srdečního tepu či nadmořské výšky v průběhu aktivity.

Data získaná výše zmíněným způsobem je nutné dále rozebrat a fragmentovat. Dalším krokem je výběr vhodné datové struktury pro uchování importovaných dat, snadný přístup k nim a rychlé a taky jednoduché vyhledávání. Pro pozdější analýzu sportovních dat je dobré si data vhodným způsobem předzpracovat již při ukládání do datové struktury, v tomto případě do předem připravené databáze. Vhodný návrh struktury databáze je v daném kroku klíčový a může velmi usnadnit analýzu sportovních výkonů uživatelů aplikace.

V dnešní době zejména čeští trenéři a jejich svěřenci nebo celé oddíly používají ručně psané tréninkové deníky. Atleti buď po dokončení tréninku, nebo třeba na konci týdne zapi-

sují jednotlivé aktivity, běhy a tréninky do sešitů, které následně odevzdají svým trenérům ke kontrole a analýze. Trenéři v návaznosti na to vytváří tréninkové plány a tréninkové kalendáře na další období. V mnoha případech je možné setkat se s tím, že využívají Microsoft Excel[1] tabulky či jiné pokročilejší programy, které však nejsou příliš vhodné pro záznam či analýzu sportovních dat. Problematických aspektů je zde hned několik. Ačkoliv mohou být trenér i jeho svěřenci proškoleni pro práci s tabulkovými procesory (Microsoft Excel, Google tabulky[2] a jiné), s vysokou pravděpodobností se jim nikdy nepodaří analyzovat sportovní data tak podrobně a sofistikovaně, jako pomocí metod pro dolování dat. Jejich využití otevírá trenérům a atletům zcela nové obzory. Dalším úskalím může být i to, že tabulkové procesory a ručně psané tréninkové deníky nejsou vhodné pro uchování velkého objemu historických dat. Ať už je zavedena sebelepší struktura, časem se tyto záznamy stávají velmi nepřehlednými. O něco lepší je situace v týmech či oddílech, kde svěřenci používají moderní sportovní aplikace, jako je například Strava (viz 5.1), Garmin Connect (viz 5.2) od výrobce sportovních hodinek a další. Jedná se o přehlednou formu uchování a zobrazení sportovních aktivit s možností historie a snadného vyhledávání. Je možné procházet aktivity několik let staré a dívat se na jejich podrobný rozbor. Nevýhodou však je, že tyto webové a mobilní aplikace neobsahují podporu týmů a pokud ano, tak ve velmi omezené podobě. Podpora trenérů zde většinou chybí úplně.

Z těchto a dalších důvodů vznikl nápad vytvořit komplexní on-line tréninkový deník, který spojuje výhody již existujících řešení, jako jsou uchování velkého množství historických dat, přehlednost, snadné vyhledání aktivit, dostupnost odkudkoliv, kde je internetové připojení a tak dále, a zároveň přináší možnost podrobnější a sofistikovanější analýzy tréninkových dat sportovců pomocí dolování dat, podporu týmů a podporu trenérů.

[1] Microsoft Excel - https://www.microsoft.com/cs-cz/microsoft-365/excel
[2] Google Sheets - https://www.google.cz/intl/cs/sheets/about

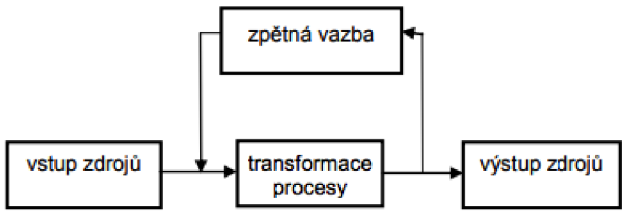
Kapitola 2

Webové aplikace a informační systémy

V následujících odstavcích budou přehledně uvedeny a stručně popsány základní pojmy týkající se informačních systémů a webových informačních systémů či aplikací. Každý pojem bude zjednodušeně popsán a vysvětlen, u některých bude uveden konkrétní příklad.

2.1 Systém

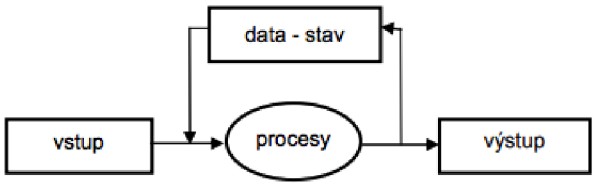
Systém je možné chápat jako konečnou množinu prvků a vazeb mezi nimi, které jsou účelově definovány na nějakém nosiči. Nosičem je množina prvků systému ve vzájemných informačních a procesních vztazích. Prvky nosiče je možné nazvat též jako zdroje. Systém se skládá ze vstupní části, prostřednictvím které zdroje vstupují do systému a výstupní části, prostřednictvím které vystupují. Mezi těmito částmi jsou transformační procesy, které přetváří vstupní zdroje na výstupní, přičemž zde probíhají různé algoritmy. Procesy jsou určitým způsobem definovány, nějak intereagují a mohou probíhat paralelně. Součástí systému může být také zpětná vazba, která ovlivňuje provádění transformačních procesů. Ne vždy je tedy vstup procesu závislý pouze na aktuálním vstupu systému, nýbrž také na jeho stavu. Stavem systému zpravidla rozumíme zdroje. U nich nás zajímá jejich perzistence (přetrvání) a konzistence (splnění daných pravidel). Zdroje můžeme dělit na fyzické, jako třeba materiál, osoby či stroje, a konceptuální, což jsou třeba informace modelující systém ve formě dat. V této práci figurují zejména konceptuální zdroje systému. [7]



Obrázek 2.1: Obecné schéma systému [7]

2.2 Informační systém

Informační systém je obyčejně systém pracující s konceptuálními zdroji. Opět se zde nachází vstupní zdroje a výstupní zdroje, kudy se do systému vkládají respektive ze systému vystupují data, která jsou pak dále interpretována jako infromace. Mezi vstupem a výstupem opět probíhá, stejně jako u obecného systému, transformace, kde procesy provádí nad vstupními daty různé algoritmy. Stav informačního systému bývá většinou reprezentován prostřednictvím dat. Přítomna je i zpětná vazba, a tedy ne všechny výstupy procesů jsou přímo závislé na okamžitých vstupech, ale typicky závisejí na již zmiňovaném stavu systému. Informační systém bývá modelem konkrétního systému reálného světa pracující s obrazy skutečných zdrojů. Jestliže v reálném systému existuje nějaká transformační funkce nad zdroji, pak v informačním systému, který tento systém modeluje, existuje obraz této funkce pracující s obrazy zdrojů. Kupříkladu systémem může být sklad nějaké firmy včetně všech prostor, expedičních ramp, administrativních archů a dalších prvků. Informačním systémem nad tímto systémem by byl software pro řízení skladových zásob příjmů a výdejů. [7]



Obrázek 2.2: Schéma informačního systému [7]

2.3 Databáze

Pod pojmem databáze je možné chápat nejen datovou strukturu pro uchování perzistentních dat, ale také software určený pro přístup k těmto datům, manipulaci s nimi a také pro správu a protokolování. Takový software se nazývá systém řízení báze dat. Databází existuje několik typů. Obyčejně bývají rozděleny dle přístupu k datům, způsobu uchování dat a dle vztahů mezi entitami. Databáze jsou relační, objektové, objektově relační, temporální, hierarchické, prostorové a další. V této práci je využívána relační databáze, která poměrně efektivně splňuje všechny požadavky na aplikaci. Relační databáze je stručně popsána v následujících odstavcích. [7]

2.3.1 Relační model dat

Relační model dat je vytvořen nad matematickou strukturou zvanou relace. Jedná se o jednoduchou a přirozenou matematickou strukturu, podle které nese název i tento model. V relačních databázích se relacím říká tabulky. Pro tuto práci není detailnější a rozsáhlejší popis relací potřeba, avšak tento kvalitní matematický základ znamená, že jednotlivé relační výrazy mohou být analyzovány, optimalizovány a mimo jiné i transfromovány na jim ekvivalentní. Relace se skládá ze schéma relace a jeho výskytu.

Výskyt relace je již výše zmiňovaná tabulka s řádky a pojmenovanými sloupci. Řádky v relaci nazýváme uspořádané n-tice, přičemž všechny řádky dané relace mají stejný počet sloupců, stejnou aritu, a zároveň žádné dvě n-tice nejsou totožné. Výskyt relace je tedy podmnožina množiny kartézského součinu nad doménami datových typů sloupců. Sloupce v tabulkách jsou pojmenované a tato jména se nazývají atributy. Hodnota atributu je prvkem množiny, která se nazývá doména daného atributu.

Id	Name	Address	Status
111111111	John Doe	123 Main St.	Freshman
666666666	Joseph Public	666 Hollow Rd.	Sophomore
111223344	Mary Smith	1 Lake St.	Freshman
987654321	Bart Simpson	Fox 5 TV	Senior
023456789	Homer Simpson	Fox 5 TV	Senior
123454321	Joe Blow	6 Yard Ct.	Junior

Obrázek 2.3: Příklad jednoho z možných výskytů relace Student [7]

Schéma relace jsou metadata výskytu relace a skládají se ze jména relace, jména atributů v relaci s jejich přiřazenými jmény domén a z integritních omezení, což jsou restrikce na výskytech relace ve schématu. [7]

```
create table Student (
   Id          integer
   Name        string
   Address     string
   Status      string
   )
```

Obrázek 2.4: Příklad metadat výskytu (schéma relace) relace Student [7]

2.3.2 Relační databáze

Relační databázi můžeme chápat jako konečnou množinu relací. Jak bylo uvedeno výše, relace jsou data a také metadata. Ne jinak tomu je u relačních databází. Množina všech metadat se v relačním modelu nazývá schéma relace. V relačních databázích to je databázové schéma. Výskyt relace, v tomto případě výskyt databáze, je možné zjednodušeně označovat jako databáze. Jednotlivé relace jsou databázové tabulky obsahující atributy, tedy sloupce. Každý záznam je reprezentován řádkem v příslušné tabulce. Jednoznačnému identifikátoru řádku tabulky se říká primární klíč, přičemž to může být buď hodnota jednoho sloupce, nebo více sloupců zároveň. Vztahy mezi tabulkami (relacemi) jsou realizovány pomocí cizích klíčů, kde cizí klíč záznamu v jedné tabulce odkazuje na primární klíč záznamu v jiné nebo i stejné tabulce. To, aby záznam, na který je odkazováno, skutečně existoval, zajišťují integritní omezení. Integritní omezení zajišťují spoustu dalších věcí, jako například unikátnost záznamů v tabulce. [7]

2.4 Web

Web je možné chápat jako nástroj pro sdílení různých druhů informací, dokumentů, souborů, zdrojů a dalších věcí prostřednictvím sítě internet mezi uživateli. Informace se zde nachází

v podobě textu, obrázků, videí, zvukových záznamů a tak podobně. Zároveň web slouží jako spojení mezi daty uloženými na serveru a klientskými aplikacemi, jako jsou třeba prohlížeče. Z počátku byl web využíván primárně pro výzkumné a studijní účely, přičemž obsah byl tvořen zejména statickými textovými dokumenty v jazyce HTML[1]. V průběhu své existence prošel mnohými změnami od Web 1.0. až po Web 3.0. Nejedná se o změny v použitých technologiích, ale o změny, jakými jsou webové stránky tvořeny a používány. Obsah Webu 1.0. tvořily výše zmíněné statické stránky. Od tohoto přístupu se postupně přešlo k Webu 2.0., který slouží ke sdílení zdrojů a společné tvorbě obsahu, což přispělo k dynamickému rozvoji on-line marketingu, internetových obchodů, sdílených úložišť a dalších, v dnešní době již velmi dobře známých nástrojů. Hlavním nástrojem, který dokonale reprezentuje tento pokrok, je Cloud a využívání cloudových služeb a úložišť. V dnešních dnech se již objevuje pojem Web 3.0., který však zatím nemá jasně daný význam. Pravděpodobně se bude jednat o sémantický web s vyšším využitím sdílení, umělé inteligence (zejména pro personalizaci obsahu) a sofistikovanější interakcí s dynamickými aplikacemi.

Nejčastějšími protokoly používanými na webu jsou HTTP (Hypertext Transfer Protocol)[2] a jeho zabezpečená verze HTTPS (Hypertext Transfer Protocol Secure) určená pro komunikaci se servery a přenos souborů ve formátu HTML, XML[3] a podobně. Další časté protokoly jsou například FTP (File Transfer Protocol)[4], DNS (Domain Name System)[5], SMTP (Simple Mail Transfer Protocol)[6] a LDAP (Lightweight Directory Access Protocol)[7]. [8]

Obrázek 2.5: Schéma zobrazující dílčí části Web 2.0

[1]HTML (Hypertext Markup Language) - https://www.w3schools.com/html
[2]HTTP - https://developer.mozilla.org/en-US/docs/Web/HTTP
[3]XML (Extensible Markup Language) - https://www.w3schools.com/xml/xml$_w$*hatis.asp*
[4]FTP - https://www.investopedia.com/terms/f/ftp-file-transfer-protocol.asp
[5]DNS - https://ns1.com/resources/dns-protocol
[6]SMTP - https://www.geeksforgeeks.org/simple-mail-transfer-protocol-smtp
[7]LDAP - https://ldap.com

2.5 Webové aplikace a webové informační systémy

Webová aplikace je aplikace, program či komplexní sofware běžící typicky na nějakém webovém serveru. Zde je tato aplikace fyzicky uložena a spuštěna. Webové prohlížeče na strojích uživatelů komunikují s daným serverem kupříkladu prostřednictvím protokolů HTTP nebo HTTPS. Jedná se často o síťovou architekturu klient-server, přičemž klientské prohlížeče zasílají požadavky na server identifikovaný pomocí IP adresy (obvykle přeložené z doménového jména prostřednictvím DNS) a konkrétní spuštěnou aplikaci určenou číslem portu (v tomto případě webový server, nebo zkráceně webserver) a přijímají a zpracovávají odpovědi. Webová aplikace uložená na tomto serveru je komplexní program, který uživatelům nebo dalším zařízením poskytuje nějakou službu, nebo řeší nějaký jejich problém. V současnosti jsou takové aplikace psány prostřednictvím programovacích jazyků jako PHP, JavaScript, Python, Java, Ruby a další. Ve velké míře jsou zde použity také značkovací jazyky HTML, XML a jiné. Hojně se využívá také JSON formát souborů. Typické je napojení na databázi, ať už se jedná o MySQL[8], MariaDB[9], Elasticsearch[10], nebo nějakou NoSQL[11] databázi. Pro přehledný a efektivní návrh a pozdější implementaci webových aplikací se využívá nejeden framework. Příkladem může být Laravel[12], Nette[13] a Django[14]. Využití zde nalezne spusta návrhových vzorů, objektově orientovaných paradigmat a architektur. Zjednodušeně je možné webovou aplikaci rozdělit na frontend a backend části (obě jsou stručně popsány v následujících odstavcích). Sofistikovanější rozdělení je možné učinit pomocí některé z dostupných architektur pro vývoj webových aplikací. Stručný popis MVC architektury použité v této práci je uveden níže (viz sekce 2.9). Webový informační systém je informační systém vyvinutý pro webové prostředí ve stylu webové aplikace.

2.6 Frontend

Pod pojmem frontend si můžeme představit prezentační část webové aplikace. Hlavní rolí je prezentace výsledků práce aplikace a vyobrazení jejího stavu. Důležitá je také interakce s uživatelem aplikace, nebo dalším navazujícím externím systémem. Zjednodušeně řečeno, frontend je to, co uživatel aplikace vidí, na co může klikat a kam může psát. Jedná se o různé výpisy, textové odstavce, seznamy, tabulky, formuláře, obrázky, videa, tlačítka, nabídky a další prvky. Důraz je zde kladen na uživatelskou přívětivost (viz sekce 2.8) a efektivní práci s aplikací nebo systémem. Frontend bývá samostatná část, která komunikuje s částí backend, nebo se obě části mohou i částečně prolínat. Technologie používané pro frontend jsou třeba HTML, JavaScript, CSS, nebo některý JavaScript framework jako jQuery[15], Vue.js[16], React[17] a jiné.

[8] MySQL - https://www.mysql.com
[9] MariaDB - https://mariadb.org
[10] Elasticsearch - https://www.elastic.co
[11] NoSQL - databáze s jinými prostředky, než jsou tabulková schémata relačních databází
[12] Laravel PHP framework - https://laravel.com
[13] Nette PHP framework - https://nette.org
[14] Django Python framework - https://www.djangoproject.com
[15] jQuery - https://jquery.com
[16] Vue.js - https://vuejs.org
[17] React - https://reactjs.org

2.7 Backend

Backend je oproti frontend části ta část, která obsahuje veškerou logiku webové aplikace. Provádí operace nad databází, reakce na uživatelské vstupy, výpočty výstupů a jiných údajů. Backend má na starosti také zabezpečení celé aplikace. Například u webového informačního systému vidí uživatel nějakou tabulku. O samotný vzhled a vykreslení tabulky se stará frontend. Backend představuje to, jakým způsobem byla data zobrazená v tabulce vybrána z databáze, souboru, nebo jiného zdroje a jak byla tato data upravena. Programovací jazyky typické pro backend webových aplikací a webových infromačních systémů jsou PHP, JavaScript, Python, Ruby, Java a dokonce to může být i C++.

2.8 Uživatelské rozhraní

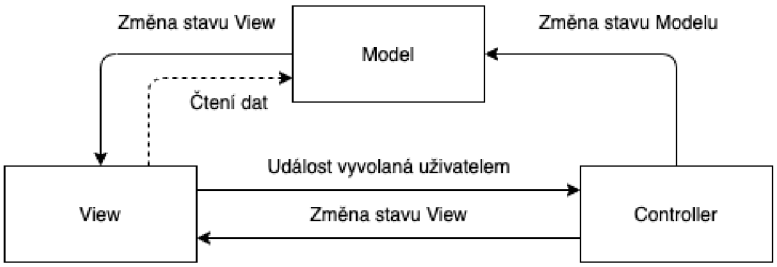
Uživatelsky orientované rozhraní je takové, jehož prvky a elementy jsou přizpůsobeny uživatelům aplikace či systému. Jednotlivé prvky nemusí přímo korespondovat s funkčními částmi systému nebo zařízení. Uživatel je tedy většinou kompletně odstíněn od toho, co se děje na pozadí, nebo co a jak funguje. Tato skutečnost uživateli usnadní a velmi zjednoduší práci se systémem. Uživatel systému nemusí vědět, že zde probíhají nějaké databázové transakce, řadící algoritmy a složité výpočty a může se plně soustředit na svoji práci. Vše v předchozí větě zmíněné je skryto, a naopak jsou ve srozumitelné podobě prezentovány výsledky průběhu procesů dané aplikace. Tím zůstávají skryty složité programové konstrukce a pro běžného uživatele odpadá nutnost jejich porozumění.[20]

2.9 MVC architektura

Vývoj desktopových i webových aplikací jde stále kupředu. V dnešní době panuje důraz na zjednodušení a zefektivnění vývoje takových aplikací. K tomu velmi dopomáhá využití objektově orientovaného programování (paradigma). Prostřednictvím tohoto paradigma se systém jeví jako soubor objektů, například instancí různých tříd, které spolu navzájem komunikují prostřednictvím zasílání zpráv. Využívá se zde mnohotvárnosti, dědičnosti a dalších vlastností. Vývoj složitějších systémů či aplikací si lze v současnosti bez objektově orientovaného návrhu a programování jen stěží představit.

MVC architektura přímo vychází z objektově orientovaného návrhu a snaží se zefektivnit návrh a následný vývoj komplexních infromačních systémů či aplikací. Jedná se o třívrstvou architekturu složenou z částí Model (datová vrstva), View (prezentační vrstva) a Controller (řídicí vrstva). Dochází tak k logickému oddělení dat, vzhledu a řízení aplikace. Největší využití nachází tato architektura při vývoji dynamických a interaktivních webových aplikací. Model zde figuruje na datové vrstvě a reprezentuje samotná data. Například jedna instance objektu Model může představovat jeden řádek v příslušné tabulce v databázi. Model může také provádět předpřipravení dat pro pozdější zobrazení. Controller realizuje řízení ke změně stavu modelu. Řídí tok aplikace a je jakousi pomyslnou spojkou mezi vrstvami Model a View. Vrstvy Model i Controller jsou při rozdělení aplikace na backend a frontend součástí backend vrstvy. Naopak View je součástí frontend vrstvy a je zprostředkovatelem prezentace dat uživateli nebo dalšímu programu.

MVC architektura umožňuje efektivní a přehledný vývoj dynamických webových aplikací a informačních systémů, proto byla využita při tvorbě této práce. Mezi reálné implementace MVC architektury (framework) pro vývoj webových aplikací patří například



Obrázek 2.6: Zjednodušené schéma architektury MVC

Symfony[18], Laravel, Nette (všechny PHP), Django (Python) a Ruby on Rails[19] (Ruby). [17]

2.10 Základní sportovní pojmy

V následujících několika odstavcích budou uvedeny a velmi stručně popsány základní sportovní pojmy použité v této práci, přičemž zde bylo čerpáno mimo jiné z knihy Training Essentials for Ultrarunning od světoznámého trenéra Jasona Koopa (viz [9]).

2.10.1 Klidový srdeční tep

Klidový srdeční tep představuje hodnotu srdečního tepu ve chvíli, kdy je sportovec v úplném klidu. To znamená, že neabsolvuje žádnou zátěž, ani není vystaven stresové situaci. Ideálně například sedí či leží. Čím trénovanější jedinec, tím bývá jeho klidový srdeční tep nižší, nemusí to však být pravidlem. [12]

2.10.2 Maximální srdeční tep

Maximální srdeční tep nastává ve chvíli maximálního vypětí sportovce, respektive ve chvíli, kdy podstupuje maximální zátěž. Tato hodnota se dá zjistit různými zátěžovými testy na běžeckém pásu nebo cyklotrenažéru. [12]

2.10.3 Laktátový práh

Laktátový práh představuje takovou hodnotu srdečního tepu sportovce, kdy spotřeba energie překonává schopnost aerobního systému tuto energii dodávat, respektive vyrábět s využitím přijímaného jídla a kyslíku. [12]

2.10.4 VO2max

Hodnota VO2max určuje maximální aerobní kapacitu jedince. Jedná se o maximální objem kyslíku, který dokáže sportovec přijmout, transportovat a využít. Většinou se uvádí v mililitrech na kilogram tělesné váhy za minutu, tedy ml/kg/min. Hodnoty se mohou různit, přičemž dobře trénovaní muži dosahují hodnot mezi 55 až 60 a ženy mezi 50 - 55. Elitní olympijští atleti mohou dosahovat až hodnoty okolo čísla 80. [10] [12]

[18]Symfony PHP framework - https://symfony.com
[19]Ruby on Rails framework - https://rubyonrails.org

Kapitola 3

Import dat do informačního systému

V této kapitole bude popsáno, jakým způsobem může probíhat komunikace mezi zařízeními či aplikacemi, na které atleti a obecně sportovci zaznamenávají svoje sportovní aktivity a tréninky, a vytvářeným on-line tréninkovým deníkem, což je v podstatě webový informační systém (viz sekce 2.5). Nejprve je nutné si ujasnit, jaká zařízení sportovci používají. Zde našel využití dotazník s několika stručnými otázkami, pomocí něhož bylo možné určit hlavní skupiny uživatelů a jejich zařízení, a na tyto skupiny tvorbu aplikace cílit. Následně je potřeba zjistit, jaký formát je k přenosu těchto dat ideální. Klíčovou roli zde hraje možnost získání a přenosu co největšího množství dat o aktivitě, aby žádné klíčové aspekty sportovního výkonu nezůstaly upozaděny. Neméně důležitá je možnost tato data snadno procházet, segmentovat a vhodným způsobem ukládat do interní databáze aplikace. Před samotným uložením do databáze je dobré data nějakým způsobem upravit, zjednodušit, vzorkovat a také předzpracovat pro další úkony a dolování dat. Tímto krokem může dojít ke zjednodušení a zrychlení následného dolování, nebo samotného zobrazení přehledů, statistik a detailů sportovních aktivit.

3.1 Nejčastěji používaná zařízení

Jak již bylo uvedeno výše, nejprve je nutné zjistit, jaká zařízení sportovci používají pro záznam svých aktivit, běhů a sportovních dat obecně. Pro začátek můžeme vyloučit sportovce, kteří si své aktivity nijak nezaznamenávají a běhají či sportují takzvaně na pocit. V tomto případě není nijak možné získat jejich sportovní záznamy, importovat je do systému a provádět nad nimi další analýzu. V některých aplikacích lze zadávat absolvované tréninky ručně. Uživatel zde zadá například uběhnutou vzdálenost, čas a subjektivní pocit náročnosti. Z těchto dat může aplikace vypočítat tempo a několik dalších zajímavých údajů. Problémem je, že úpně chybí záznamy pro vývoj srdečního tepu, tempa, rychlosti, nadmořské výšky a další důležité aspekty z průběhu aktivity. Nad těmito ručními či manuálními záznamy není možné provádět dolování dat ani pokročilou analýzu, protože pro to jednoduše chybí data.

Pokud je opomenuta výše uvedená skupina sportovců, kteří sportují bez zařízení, které by jejich aktivitu zaznamenávalo, a tedy aplikace vyvíjená v této práci pro ně nemá význam, zbydou dvě skupiny uživatelů. Jedni využívají při sportovních aktivitách chytrý telefon, ti druzí sporttester nebo chytré hodinky. Obě poslední uvedené technologie jsou více méně

podobné. Sportovci, kteří svou aktivitu zaznamenávají na chytrý telefon, často využívají některou z volně dostupných, nebo i placených sportovních aplikací k tomu určených. Podle článku [13], který vyšel zhruba před rokem na běžci uznávaném webu Runner's World (vychází od roku 1966), jsou to následující mobilní aplikace, dostupné jak pro operační systém Android od firmy Google, tak pro iOS od firmy Apple:

- Human[1],
- Coach to 5k[2],
- Pacer[3],
- Strava (viz 5.1),
- MyCoach[4],
- MapMyRun[5],
- Nike+ Run Club[6] a další.

V českém prostředí může výše uvedené aplikace doplnit ještě ASICS Runkeeper[7] a EPP - Pomáhej pohybem od nadace ČEZ[8]. Všechny výše zmíněné aplikace fungují dobře zejména pro začátečníky. Nabízí měření vzdálenosti, vývoj tempa v průběhu aktivity a další vymoženosti. Některé dokáží vykreslit i profil trasy a vývoj nadmořské výšky. Tento údaj bývá však nepřesný, jelikož aplikace nevyužívají barometr, nýbrž nadmořskou výšku počítají z mapy. Dále chybí i záznam srdečního tepu. Z těchto a dalších důvodů nejsou aktivity zaznamenané ve sportovních aplikacích chytrých telefonů využitelné pro tuto práci. Opět je zde absence dostatečně velkého množství kvalitních a spolehlivých dat pro dolování a analýzu. Nicméně i tak lze takto zaznamenané tréninky importovat do vyvíjeného systému. Tato práce je však spíše zaměřená na uživatele chytrých hodinek a sporttesterů.

Spousta atletů a výkonnostních sportovců v dnešní době používá pro záznam tréninků sporttester nebo chytré hodinky. Zároveň tito sportovci většinou trénují pod vedením trenéra či více trenérů, nebo jsou součástí nějakého klubu nebo sportovního oddílu. Zejména tato skupina uživatelů je vzhledem k uplatnění a využitelnosti vyvíjené aplikace tou cílovou. Nicméně i amatérští a hobby sportovci již dnes přecházejí od mobilních aplikací ke sporttesterům. Jedná se o pohodlnější variantu záznamu aktivity, která nabízí možnost podrobnějších statistik a zároveň je dnes již poměrně cenově dostupná. Pro tuto práci je důležitý fakt, že sporttestery dokážou zanamenat velmi přesně vzdálenost aktivity, nadmořskou výšku, tep, tempo, rychlost, spálené kalorie a další údaje. Zároveň umí tyto údaje sbírat v čase, tedy v průběhu celé aktivity. Jako příklad je možné uvést vývoj hodnoty srdečního tepu a nadmořské výšky. Je možné tedy zjistit, že sportovec měl na 5. kilometru srdeční tep 145 úderů za minutu a zároveň v tuto chvíli běžel do kopce. Toto jsou velmi užitečné informace, ze kterých lze dopočítat a extrahovat zajímavé skutečnosti a fakta. Zde

[1] Human - http://human.co
[2] Coach to 5k - https://play.google.com/store/apps/details?id=com.phe.couchto5Khl=en$_GBgl = US$
[3] Pacer - https://www.mypacer.com
[4] MyCoach - https://play.google.com/store/apps/details?id=com.globalsport.prohl=csgl=US
[5] MapMyRun - https://www.mapmyrun.com
[6] Nike+ Run Club - https://www.nike.com/cz/nrc-app
[7] ASICS Runkeeper - https://runkeeper.com/cms
[8] EPP - https://pomahejpohybem.cz

již nachází uplatnění dolování dat pro podrobnou analýzu sportovních záznamů s možností například určení typu běžce.

V předchozích odstavcích je odůvodněno, proč je tato aplikace (on-line tréninkový deník) zaměřena zejména na uživatele sporttesterů a chytrých hodinek. V dalším kroku je však nutné zjistit a analyzovat, jaké sporttestery a hodinky sportovci preferují. K tomu posloužil vytvořený dotazník.

3.1.1 Dotazník

Dotazník byl vytvořen za účelem zjištění, jaké sportovní hodinky sportovci používají. Od toho se pak dále může odvíjet výběr technologií pro přenos záznamů o aktivitách. Dotazník byl realizován prostřednictvím Google Dotazníku a byl složen ze dvou jednoduchých otázek. První otázka byla „Jakou technologii používáte pro záznam sportovních aktivit?“. Zde byly na výběr tři možnosti, a to:

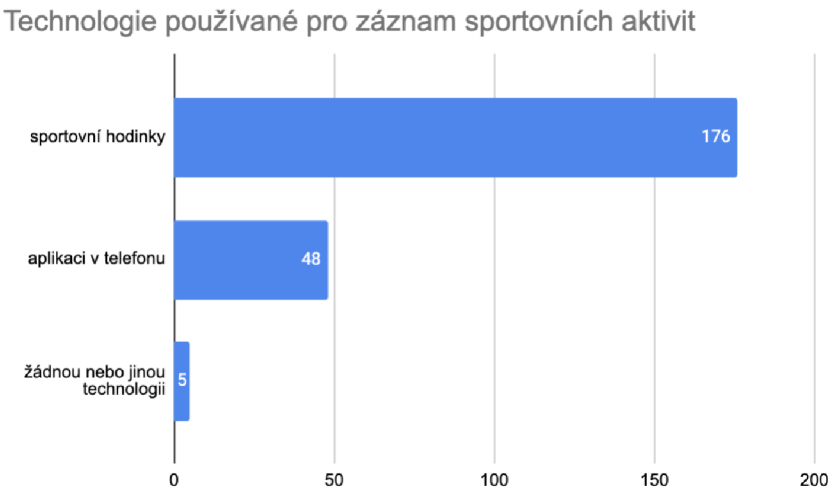
- sportovní hodinky (sporttester nebo chytré hodinky),
- sportovní aplikaci v chytrém telefonu a
- žádnou nebo jinou technologii.

Druhá otázka zněla „Jestliže používáte pro záznam sportovní aktivity chytré hodinky (nebo sporttester), jaké značky jsou?“ s možnostmi:

- Samsung,
- Apple,
- Suunto,
- Garmin,
- Polar,
- Xiaomi,
- Fitbit,
- Coros a
- jiné.

Tento dotazník byl ručně vyplněn s několika sportovci, dále byl rozeslán atletům prostřednictvím služby Messenger a umístěn na několik bežecky a sportovně orientovaných skupin na platformě Facebook. Celkem odpovědělo 238 respondentů. Po odstranění odpovědí, kde například dotazovaný uvedl, že používá sportovní aplikaci v chytrém telefonu a zároveň zatrhl, že používá sportovní hodinky od určitého výrobce a podobně, zbylo 229 vypovídajících odpovědí. Trend odpovědí je možné vidět v následujících grafech (**obrázky 3.1 a 3.2**).

Z prvního grafu je možné vyčíst, že 176 respondentů na otázku „Jakou technologii používáte pro záznam sportovních aktivit?“ odpovědělo, že používají sportovní hodinky, 48 respondentů uvedlo jako odpověd aplikaci v telefonu a dalších 5 dotazovaných nepoužívá žádnou, nebo jinou technologii, než která byla v nabídce. V přepočtu na procenta to činí



Obrázek 3.1: Graf odpovědí na otázku: Jakou technologii používáte pro záznam sportovních aktivit?

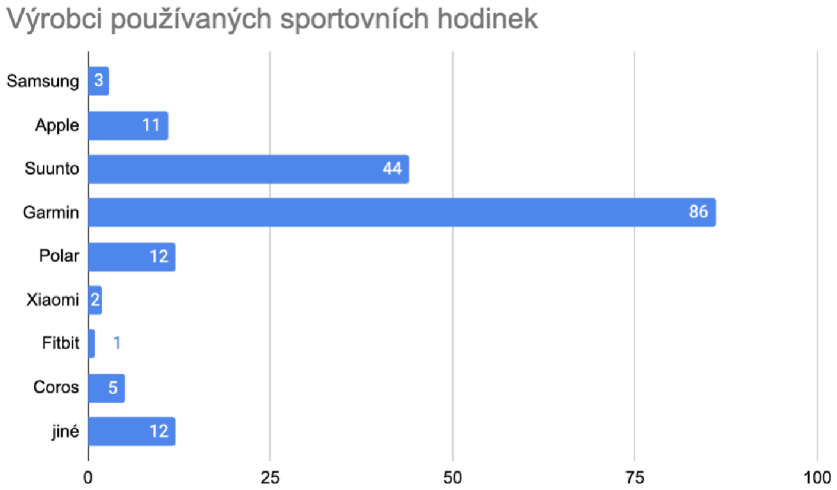
zhruba 77 % pro sportovní hodinky, 21 % pro mobilní aplikaci a přibližne 2 % pro žádnou, nebo jinou technologii. V tomto grafu je možné pozorovat trend upřednostňování sportovních hodinek, sporttesterů a chytrých hodinek na úkor sportovních aplikací v chytrých mobilních telefonech a jiných technologií.

V druhém grafu je možné vidět, že nejvíce respondentů využívajících pro záznam svých aktivit sportovní hodinky, vlastní sportovní hodinky značky Garmin. Jedná se o 86 případů z celkových 176, což dělá přibližně 49 %. Druhým nejpoužívanějším výrobcem je finská společnost Suunto. Suunto zvolilo 44 dotazovaných, a tedy přibližně 24,5 %. Následují v tomto pořadí Polar (12 odpovědí), Apple (11 odpovědí), Coros (5 odpovědí), Samsung (3 odpovědi), Xiaomi (2 odpovědi) a Fitbit (1 odpověď). Celkem 12 respondentů vybralo možnost jiné.

Prostřednictvím dotazníku bylo možné získat povědomí o využívání technologií pro záznam aktivit v části sportovní komunity a také povědomí o rozšíření výrobců sportovních hodinek mezi sportovci. Zjednodušeně řečeno, většina dotazovaných sportovců používá sportovní hodinky, a to zejména od výrobců Garmin a Suunto, přičemž někteří mají i Polar a Apple hodinky. Z těchto poznatků je možné vycházet při návrhu a realizaci importu sportovních dat do systému. Dotazník se nachází v příloze A.

3.2 Možnost tvorby aplikací pro chytré hodinky

Jedním z bodů zadání bylo seznámení se s tvorbou aplikací pro chytré hodinky a možnostmi komunikace webových aplikací s chytrými hodinkami různých značek. Jak bylo uvedeno v předchozích odstavcích této kapitoly, většina dotázaných sportovců v dnešní době využívá pro záznam aktivit sportovní hodinky či sporttestery a to zejména značek Garmin, Suunto a Polar. Tito výrobci buď vůbec nepodporují instalaci aplikací třetích stran do svých zařízení, nebo jen ve velmi omezené podobě (například widget na hlavní obrazovku). Pravidlem ale bývá, že uživatelé zmíněných zařízení mají možnost nahrávat aktivity na servery výrobců, odkud je mohou exportovat v různých formátech. Z tohoto důvodu bylo usouzeno,



Obrázek 3.2: Graf odpovědí na otázku: Jestliže používáte pro záznam sportovní aktivity chytré hodinky (nebo sporttester), jaké značky jsou?

že nemá smysl vyvíjet aplikace přímo pro sportovní hodinky, ale je lepší získávat data o aktivitách atletů ze serverů výrobců těchto zařízení. Zmíněný postup je uveden v kapitole implementace (viz 8).

3.3 Formát pro přenos sportovních dat

Pro přenos sportovních dat, respektive záznamů o sportovních aktivitách, byl vybrán formát FIT, celým názvem Flexible and Interoperable Data Transfer, což v překladu znamená formát pro flexibilní a interoperabilní přenos dat. Jedná se o binární formát vyvinutý společností Garmin a plně podporovaný jejími zařízeními. Jak je možné vidět na grafu používaných hodinek na obrázku **3.2**, je použitím tohoto formátu pokryta největší sorta potenciálních uživatelů. Další výhodou formátu je, že s ním pracují také výrobci Suunto, Polar i Apple. U těchto výrobců je možné exportovat aktivity manuálně právě ve fromátu FIT. Dokonce i Suunto a Polar API umí vracet aktivity v binárním formátu FIT. Ten obsahuje všechna potřebná data o aktivitách jako vzdálenost, časy, tempa, srdeční tep, nadmořskou výšku a spoustu dalších informací. Díky této obsáhlosti, flexibilnosti a rozšířenosti mezi různými výrobci sportovních hodinek se jeví jako vhodný formát pro tuto práci a bude zde použit. Více informací je možné nalézt na stránkách Garmin přímo určených pro FIT SDK[9].

[9]Garmin FIT SDK - https://developer.garmin.com/fit

Kapitola 4

Získávání znalostí z databází

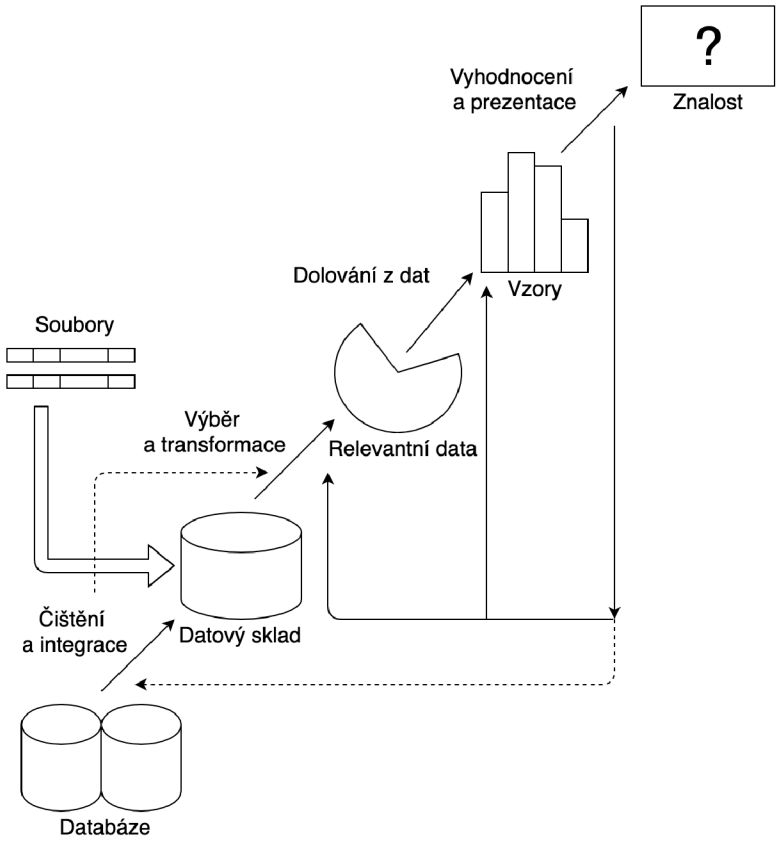
V současnosti máme k dispozici obrovské množství různých druhů dat. Jedná se o data mnoha senzorů, kamerových systémů, data z rozsáhlých informačních systémů, data získaná z webového prostředí a z mnoha dalších zdrojů. Množství dat je několikanásobně větší, než například před padesáti lety. Nárůst tohoto objemu způsobily mimo jiné nástup internetu a webového obsahu, ale i jiné moderní technologie. S tím, jak se zvyšoval objem dat, bylo nutné zajistit adekvátní metody a techniky pro zisk odpovídajících znalostí z nich. Zanedlouho přestaly stačit klasické relační, nebo jiné produkční databáze a statistické a jiné metody nad nimi použitelné. Z této nutnosti získání znalostí z velkého množství dat vznikl celý obor a celé odvětví datových skladů, dolování dat a získávání znalostí z databází. Tento obor se využívá například ve světě marketingu, oblasti analýzy trhu, věděckého bádání, odhalování podezřelých aktivit a podvodů, detekce útoků a tak podobně. Jedná se o velmi dynamicky se rozvíjející odvětví s mimořádným přesahem za hranice IT světa. [21] [6] [1]

4.1 Dolování dat a získávání znalostí z databází

Pojmy dolování dat a získávání znalostí z dat či databází jsou v dnešní době brány jako synonyma, ačkoliv je striktně vzato dolování dat pouze jedním krokem procesu získávání znalostí z dat. Setkáme se i s označením jako bagrování dat (data dredging), datová archeologie (data archeology), nebo data mining, což je anglický překlad dolování dat. Ať už je tento proces označen jakkoliv, jedná se o extraxci zajímavých modelů dat a vzorů z velkých objemů dat, přičemž jak vzory, tak modely reprezentují získané znalosti. Pod slovem zajímavých zpravidla chápeme modely a vzory netriviální, dříve neznámé, skryté a užitečné. Netriviální znamená, že danou informaci nemůžeme získat například nějakým komplikovaným SQL dotazem, ale jsou k tomu potřeba sofistikovanější metody. Skrytost znamená, že daná znalost či informace není na první pohled v datech zřetelná. Zde by nějaké komplikovanější dolování postrádalo smysl. Samozřejmě taková znalost musí být i dříve neznámá. Získávání již známých znalostí opět postrádá smysl. V neposlední řadě by taková informace měla být potenciálně užitečná pro další využití, například v marketingu pro cílení reklamy, pro odhalování nepravostí a podobně. [21] [6] [1]

4.1.1 Proces získávání znalostí

Na obrázku **4.1** je možné vidět schéma procesu získávání znalostí z dat. Jednotlivé kroky tohoto procesu budou stručně popsány v následujících odrážkách.



Obrázek 4.1: Schéma procesu získávání znalostí

1. Čištění dat - data jsou vyčištěna od šumu, jsou doplněna chybějící data a vyřešeny všechny nekonzistence.

2. Integrace dat - provede se spojení dat z několika datových zdrojů s tím, že jsou například sjednoceny názvy atributů a podobně.

3. Výběr dat - vyberou se data, která jsou vhodná a potenciálně užitečná pro dané získávání znalostí.

4. Transformace dat - transformací dat rozumíme její konsolidaci, která způsobí, že data jsou následně více vhodná pro samotné dolování. Pod pojmem transformace je možné si představit třeba agregaci nebo normalizaci.

5. Dolování dat - jedná se o jádro celého procesu, kde se pomocí konkrétní metody nebo algoritmu vytvoří model dat, případně vzor. Jedná se například o metodu shlukování pomocí algoritmu DBSCAN a další.

6. Hodnocení modelů a vzorů - identifikují se zajímavé vzory.

7. Prezentace znalostí - získané znalosti se prezentují uživateli. [21]

4.1.2 Druhy dat pro dolování

Teoreticky jsou téměř všechna data vhodná pro dolování, ať už se jedná o perzistentní data v úložištích, nebo o data transientní, která produktuje například nějaký senzor. Nejčastěji se dolují z relačních databází. Nějaká firma či společnost má kupříkladu velké množství dat o prodeji svých produktů, doplňování zboží a tak dále. Dalším příkladem může být marketingová firma, která má za úkol cílit reklamu na specifickou sortu zákazníků. Z relační databáze je sice možné získat nejrůznější zajímavé informace prostřednictvím SQL dotazů, ale to není často dostačující. Například pro shlukování produktů dle různých aspektů prodeje, nebo automatizovanou klasifikaci zákazníků dle způsobu, jakým nakupují. Tímto způsobem je možné dolováním dat získat zajímavější informace, jako jsou trendy, modely a vzory, než pouhým dotazováním pomocí jazyka SQL.

Často používaným zdrojem pro dolování dat jsou také datové sklady, do nichž se data integrují z různých databází. Mohou to být například jednotlivé databáze dceřiných společností nějaké firmy. Ta pak může pravidelně v nějakém časovém intervalu shromažďovat data do jednoho centralizovaného datového skladu, který uchovává historická data za dlouhé časové období. Data jsou zde vyčištěna a mohou být také normalizována. V každé z dceřiných databází se může struktura lišit a je potřeba určit jednotnou strukturu skladu a data podle této struktury transformovat. Datový sklad má většinou podobu multidimenzionální kostky, kde každá dimenze odpovídá atributu nebo skupině atributů ve schématu.

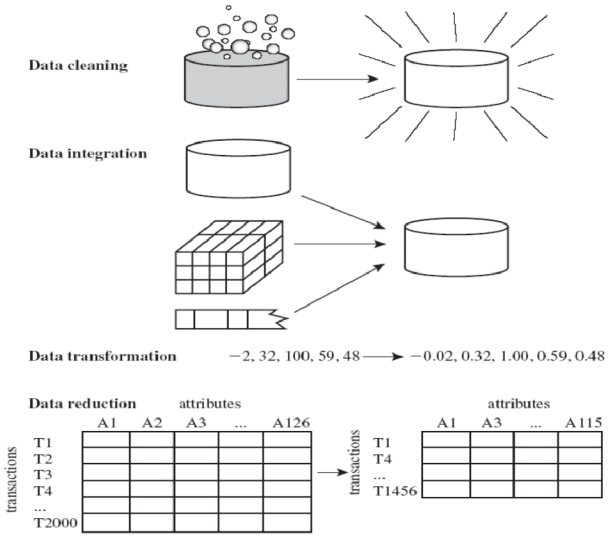
Dalšími zdroji pro dolování dat mohou být transakční databáze, objektově-relační databáze, temporální databáze, textové databáze, databáze časových řad, proudy dat, web a velká spousta dalších. [21] [6]

4.1.3 Předzpracování

Před samotným dolováním je žádoucí data určitým způsobem upravit. Pro některé záznamy mohou hodnoty sloupců chybět. Dále také může nastat nekonzistence dat při integraci do datového skladu z několika dílčích zdrojů, potažmo databází. Je možné, že hodnoty mohou být nesprávné. To bývá způsobeno třeba ve chvíli, kdy data pochází z webových formulářů nebo sběrných senzorů. Je tedy nutné určit hodnoty, které se odchylují od ostatních více, než je běžné. K tomu se využívají mimo jiné metody původem ze statistiky. Dalším problémem může být obrovské množství dat, ačkoli je to paradox. Jak je uvedeno výše, dolování dat je realizováno zejména nad velkými objemy dat. Nicméně, čím více je dat, tím více času trvá samotný algoritmus dolování. Proto je více než vhodné data redukovat tak, aby byla zachována jejich původní vypovídající hodnota. Hlavní úlohy (graficky znázorněné na obrázku **4.3**) předzpracování jsou:

- Čištění dat - během tohoto kroku se doplní chybějící data, odstraní se nekonzistence a položky, které se nachází mimo obvyklý rozsah. Pokud by dolování proběhlo nad nekvalitními daty, i výsledky by byly nekvalitní.
- Integrace dat - data se shromažďují z několika dílčích zdrojů, například databází, do jedné centrální databáze či datového skladu.
- Transformace dat - mezi transformační metody patří například agregace a normalizace.
- Redukce dat - jak již bylo zmíněno v předchozím odstavci, velký objem dat způsobuje zpomalení dolovacího algoritmu. Proto je vhodné data redukovat, ať už z hlediska počtu atributů, dimenzí, nebo hodnot.

- Diskretizace dat - jedná se o jednu z metod redukce dat. Spojité hodnoty jsou diskretizovány, a tím je redukován celkový počet hodnot. [21] [6]



Obrázek 4.2: Úlohy předzpracování dat [6]

4.2 Klasifikace a predikce

Jedním z typů metod dolování dat je klasifikace a predikce. Klasifikace umožňuje přiřazovat data do určitých tříd na základě jejich vlastností. Typickým příkladem je banka a poskytování úvěrů. Na základě údajů o novém klientovi umí tato metoda přiřadit klienta do některé z tříd, což může pomoci rozhodnout, zda klientovi úvěr poskytnout, nebo ne. Predikce naopak předpovídá spojité hodnoty objektů na základě již známých vlastností. Před započetím klasifikace a predikce je nutné data předzpracovat pomocí čištění, transformace a dalších metod. Toto je popsáno v předchozí sekci o předzpracování (4.1.3). Klasifikační i predikční metody se porovnávají dle několika aspektů, jako jsou přesnost odpovědi, rychlost, robustnost, interpretovatelnost a stabilita. [21]

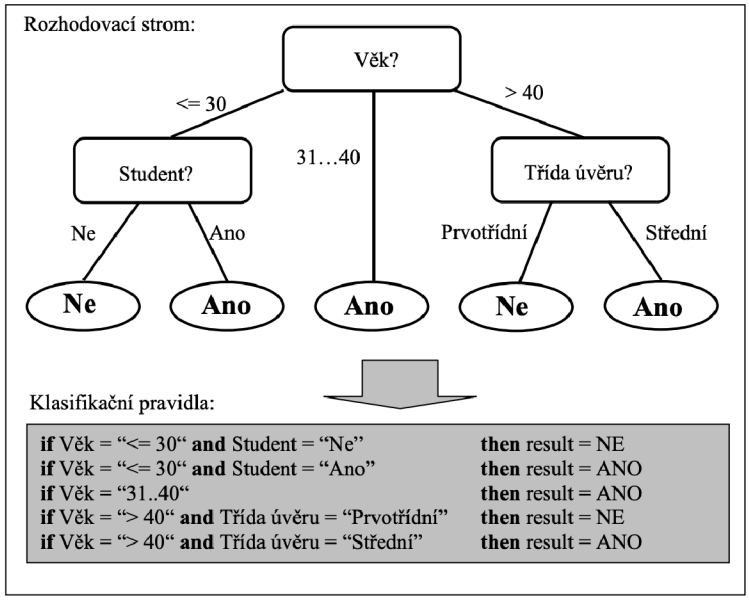
4.2.1 Klasifikační metody

V následujících několika odstavcích budou popsány klasifikační metody jako klasifikace pomocí rozhodovacích stromů, bayesovská klasifikace a klasifikátor založený na k-nejbližším sousedství. Existují však i další metody, například neuronová síť Backpropagation, SVM (Support vector machines - metoda podpůrných vektorů), klasifikátor založený na genetických algoritmech a klasifikátor založený na fuzzy množinách. [21]

Klasifikace pomocí rozhodovacích stromů

Rozhodovací strom, jak už název napovídá, je stromová struktura, kde jednotlivé uzly obsahují test hodnoty nějakého atributu a hrany jsou konkrétní hodnoty tohoto atributu. Listy jsou poté finální třídy, do kterých jsou záznamy klasifikovány. Existuje algoritmus (viz [21]),

který vytvoří pro danou databázi odpovídající rozhodovací strom. Při sestavení stromu se vždy vybere přednostně atribut (jako uzel) s nejvyšší rozhodovací schopností. Blíže kořenu stromu se tedy nachází atributy s vyšší rozhodovací schopností než atributy blíže listům. Atribut s nejvyšší rozhodovací schopností je kořenovým uzlem. Takto sestrojený rozhodovací strom je pak možné algoritmicky převést na ekvivalentní sadu klasifikačních pravidel. Rozhodovací strom a k němu ekvivalentní klasifikační pravidla je možné vidět na obrázku **4.4**. Při tvorbě rozhodovacího stromu mohou vzniknout větve, které dělají strom zbytečně složitější nebo méně kvalitní z hlediska předpovědí. Tyto větve je nutné eliminovat. K tomu slouží metody prepruning a postpruning. [21]



Obrázek 4.3: Rozhodovací strom a k němu ekvivalentní klasifikační pravidla [21]

Bayesovská klasifikace

Na rozdíl od rozhodovacích stromů je bayesovská klasifikace založena na použití statistických metod, které určí, do jaké třídy prvek pravděpodobně patří. Prvek je vždy zařazen do třídy, pro kterou vychází největší pravděpodobnost, že do ní patří. Je dán výraz P(X|Y), kde X a Y jsou nějaké různé jevy. Tento výraz říká, jaká je pravděpodobnost, že nastane jev X, když bezpečně víme, že nastal jev Y. Uvedenému výrazu se říká podmíněná pravděpodobnost nebo také bayesův vzorec a hraje klíčovou roli v bayesovské klasifikaci. Jeho celá podoba je: [21]

$$P(X|Y) = \frac{P(X \cap Y)}{P(Y)} \tag{4.1}$$

Klasifikátor založený na k-nejbližším sousedství

Klasifikátor založený na k-nejbližším sousedství je další metodou klasifikace dat do tříd. Zde je každý vzorek dat reprezentován n-ticí atributů číselné hodnoty. Z toho vyplývá, že jednotlivé vzorky jsou součástí n-dimenzionálního prostoru. Je možné si je představit jako

body tohoto prostoru, kde souřadnice jsou právě hodnoty atributů. Pro každé dva vzorky je definována euklidovská vzdálenost, kterou je možné vyjádřit následujícím vzorcem:

$$d(X,Y) = \sqrt{\sum_{i=1}^{N}(x_i - y_i)^2} \tag{4.2}$$

K dispozici jsou trénovací data, ze kterých je vybráno k nejbližších vzorků, jejichž vzdálenost je nejmenší od aktuálně klasifikovaného vzorku. Z k vybraných nejbližších vzorků se zjistí třída, která má největší zastoupení. Ta se následně přiřadí klasifikovanému vzorku. [21]

4.2.2 Predikční metody

Základní predikční metodou je lineární jednoduchá regrese, která může být zobecněna na lineární násobnou regresi. Tyto metody využívají princip nejmenších čtverců. Lineární jednoduchá regrese umí z jednoho vstupního atributu predikovat jeden výstupní. Naopak lineární násobná regrese může predikovat výsledek z několika vstupních atributů. Dalším typem je nelineární regrese používaná ve chíli, kdy nejsou vstupní a výstupní atributy lineárně závislé. [21]

4.3 Shluková analýza

Další technikou dolování dat je shlukování nebo také shluková analýza. Z pohledu strojového učení se jedná o techniku učení bez učitele. Nejsou zde zapotřebí žádná trénovací data ani předem definované třídy. Výjimkou je konceptuální shlukování, kde je předem daný koncept. Shluková analýza provádí řazení objektů do tříd na základě podobnosti. Objekty v jedné třídě by si měly být co nejvíce podobné a zároveň by neměly být příliš podobné objektům z jiných tříd. Této techniky se využívá například pro analýzu trhu, profilování a shlukování osob na sociálních sítích, shlukování podobně nakupujících zákazníků internetového obchodu, nebo také jako součást předzpracování pro jiné metody dolování dat, jako třeba pro klasifikaci. Shlukování probíhá na základě hodnot atributů datového vzorku a na jejich vzdálenostech, přičemž využívá nejrůznější poznatky ze statistiky, biologie a strojového učení.

Pro efektivní získávání znalostí z objemných databází musí shlukovací metody splňovat několik vlastností, mezi něž patří například škálovatelnost. Vzhledem k tomu, že se techniky a metody dolování dat používají opravdu nad velkým množstvím dat, musí být shlukovací metody velmi dobře škálovatelné. Tyto metody by měly být také schopné zpracovávat různé typy atributů od numerických přes binání, až po ordinální a další. Nasnadě je i schopnost vytvářet shluky různých tvarů, nejen kulovitých, za minimální znalosti daného problému. Občas bývá obtížné zadat cílový počet shluků a zároveň to může i snížit kvalitu výsledků. Některými dalšími klíčovými vlastnostmi jsou schopnost vyrovnat se s daty obsahující šum, necitlivost na pořadí vstupních záznamů a schopnost produkovat kvalitní, interpretovatelné a použitelné shluky.

Shlukovací metody nejčastěji pracují nad dvěma datovými strukturami. Jedná se o datovou matici a podobnostní matici. Shlukování obvykle probíhá nad množinou N objektů, kde každý objekt má P atributů. Datová matice je základní datovou strukturou, která má podobu relační tabulky nebo N*P matice. Oproti tomu podobnostní matice je N*N trojúhelníková matice obsahující vzdálenosti pro každé dva objekty. Vzdálenost se počítá

různými způsoby v závislosti na tom, jakého typu jsou hodnoty atributů. Tyto atributy mohou být intervalové, binární, ordinální, nominální a tak dále. Také se může jednat o proměnné různých typů. [21]

4.3.1 Typy metod shlukování

Shlukovací metody lze rozdělit do nějkolika skupin. Každá z nich může mít lepší využitelnost pro různé typy úloh. Často se využívá i kombinace metod pro porovnání výsledků. Typy shlukovacích metod jsou:

- metody založené na modelech,
- metody pro shlukování vysoce dimensionálních dat,
- metody založené na mřížce,
- metody založené na rozdělování,
- hierarchické metody a
- metody založené na hustotě. [21]

Metody založené na rozdělování

Jedná se o metody, které rozdělují N objektů do K tříd, přičemž K musí být předem známé. Platí pravidlo, že každý z N objektů je obsažen právě v jedné třídě a zároveň každá třída obsahuje alespoň jeden objekt. Všechny metody založené na rozdělování mají společné to, že na začátku je vybráno K objektů z celkového počtu N objektů. Každý z těchto objektů nyní reprezentuje třídu. Ostatní objekty jsou přiřazeny do jednotlivých tříd na základě podobnosti, potažmo vzdálenosti. Následně je v každé třídě/shluku určen nový objekt reprezentující daný shluk a ostatní objekty jsou mezi shluky různě přesouvány. Toto probíhá iterativně do okamžiku nalezení optimálních shluků. Metody založené na rozdělování vytváří kulovité shluky pro malý nebo středně velký objem dat a zároveň je nutné zadat výsledný počet shluků, tedy číslo K. To jsou jejich nevýhody. Mezi heuristiky sem spadající patří k-means a k-medoids.

K-means V metodě k-means se objekt reprezentující shluk volí jako fiktivní bod, jehož atributy vychází ze střední hodnoty atributů daného shluku. Iteruje se do té chvíle, dokud odpovídající funkce nezačne konvergovat. K tomu se využívá čtverec chyby. V praxi to znamená, že se iteruje, dokud se objekty stále ještě přesouvají mezi shluky. Nevýhody jsou výše zmíněné zadání čísla K, citlivost na šum, nemožnost nalézt shluky nekonvexního tvaru a to, že metoda nemusí najít nejoptimálnější řešení a může uváznout na lokálním minimu.

K-medoids Jedná se o metodu velmi podobnou metodě k-means. Rozdílem je to, že objekt reprezentující shluk není fiktivní, ale jedná se o reálný objekt té dané třídy, který se nachází nejblíže středu shluku. Při výběru reprezentujícího objektu se bere v potaz, jak a zda by se přesunuly ostatní objekty třídy. Algoritmus je vhodný spíše pro menší objem dat. [21]

Metody založené na hustotě

Metody založené na hustotě jsou primárně reprezentovány algoritmy DBSCAN a DENCLUE. Jedná se o metody, které považují za shluky oblasti s vysokou hustotou objektů oddělené oblastmi s nízkou hustotou objektů, které jsou považovány za šum. Tyto metody tak umí produkovat shluky různých tvarů a jsou odolné vůči šumu.

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) DBSCAN je algoritmus spojující do shluku oblasti s dostatečnou hustotou výskytu objektů. Jak velká hustota musí být, aby vznikl shluk, je zadáno jako vstupní parametr algoritmu. Algoritmus projde celou databázi a pro každý objekt určí, zda jeho ε-okolí (okolí bodu o poloměru ε) obsahuje alespoň určitý požadovaný minimální počet objektů. Pokud ano, vytvoří shluk a daný bod nazve jádrem shluku. Následně se iterativně hledají objekty, které jsou přímo dosažitelné z jádra shluku a ty jsou do shluku přidány. Tímto způsobem se spojují shluky a algoritmus končí ve chvíli, kdy nemůže být žádný další objekt přidán do žádného shluku. Nevýhodou je opět zadání vstupních parametrů.

DENCLUE (DENsity-based CLUstEring) Dalším příkladem metody založené na hustotě je algoritmus DENCLUE. Jedná se o algoritmus založený na využití distribučních funkcí hustoty. Výhodou je právě to, že je založený na matematickém základu a je možné matematicky popsat shluky jakéhokoliv tvaru i pro vysoce dimensionální data. Podrobnější informace a popis je možné nalézt zde [6].

Metody založené na mřížce

Jak již název napovídá, metody založené na mřížce využívají víceúrovňovou mřížkovou strukturu. Ta rozděluje všechny objekty do shluků. Výhodou je vysoká rychlost shlukování a nezávislost na počtu objektů. Tyto metody jsou závislé pouze na velikosti mřížky. Příkladem mohou být metody WaveCluster (založená na vlnkové transformaci) a STING (využívá statistických informací z mřížky). [21]

Kapitola 5

Existující řešení

V této kapitole budou představeny již existující implementované sportovní aplikace pro podporu tréninku a analýzu sportovních dat. Jedná se zejména o webové aplikace, které mohou mít i mobilní podporu. Aplikace budou stručně představeny, zhodnoceny, porovnány a budou zmíněny jejich hlavní výhody a nevýhody. Mezi základní kritéra hodnocení bude patřit uživatelská přívětivost, možnost zobrazení co nejpodrobnějších statistik, podpora týmů, trenérů, tréninkových plánů a zejména podpora krátkodobé, nebo dlouhodobé analýzy tréninkových dat a jejich využitelnost při vytváření trénikových plánů, nebo závodní strategie jak v rámci týmu, tak pro jednotlivce.

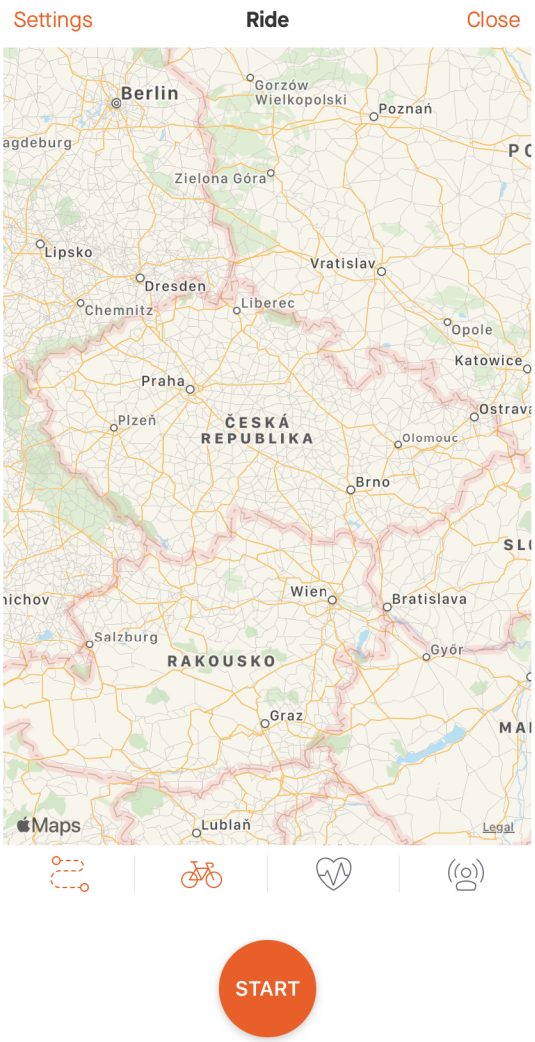
5.1 Strava[1]

Strava je jedna z nejrozšířenějších a nejznámějších sportovních aplikací současnosti. Její fungování započalo v roce 2009, kdy ji v San Franciscu v Kalifornii založili Mark Gainey a Michael Horvath. Je považována za sociální síť atletů a celkově všech sportovních nadšenců. Mimo jiné je výjimečná tím, že funguje jako Facebook nebo Instagram ve světě sportu. Uživatelé zde mají přátele a sledující, kteří mohou vidět jejich sportovní záznamy, komentovat je a udělovat jim „to se mi líbí“, takzvaná kudos. Tímto se aplikace odlišuje od ostatních, a proto je populární napříč širokou sportovní veřejností.

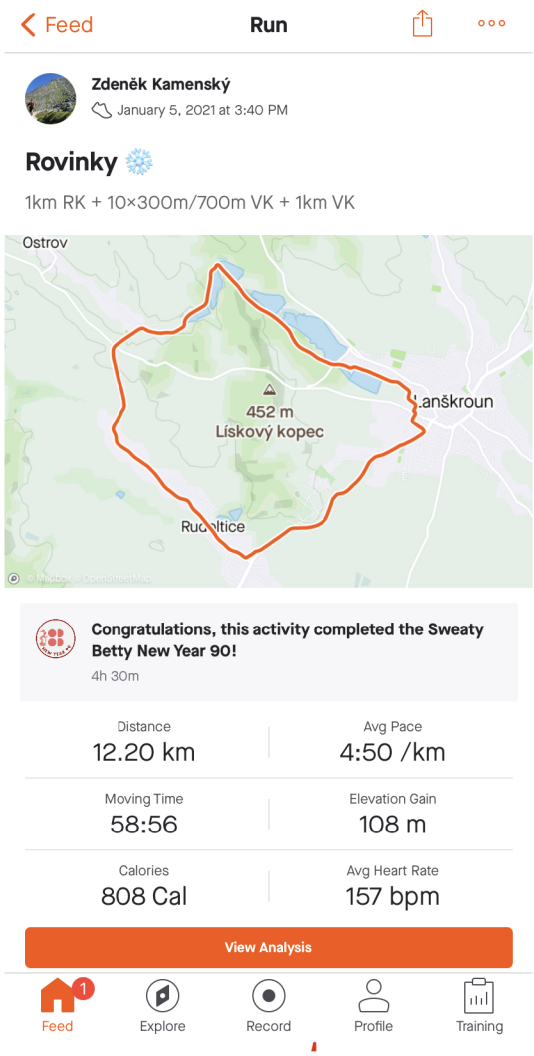
Jedná se jak o webovou aplikaci, tak také o mobilní aplikaci pro operační systémy Android a iOS. Uživatel, který si zde založí účet, má možnost využívat jak mobilní, tak webovou verzi. Existuje také placená varianta s pravidelnou měsíční taxou a ročním předplatným, která nabízí více funkcí. Mobilní verze může fungovat přímo jako záznamové zařízení. Pro sporty, kde je vyžadován pohyb z místa na místo, jako jsou běh nebo cyklistika (například cvičení v posilovně sem nepatří), je nutné mít v mobilním telefonu povoleno získávání polohy GPS. Aplikace potom umí poměrně přesně zaznamenávat aktivitu, kterou můžeme sdílet s přáteli. Na obrázcích **5.1** a **5.2** vidíme, jak je s pomocí této aplikace možné nahrávat sportovní aktivitu prostřednictvím chytrého telefonu a také následný detail aktivity.

Do aplikace lze také automaticky importovat sportovní data z několika dostupných zdrojů. Těmi jsou například Garmin, Polar, Wahoo a Zwift. Dále je také možné sportovní aktivitu nahrát ručně. Mezi další výhody patří fakt, že lze shlukovat sportovce do klubů. Zde se mohou porovnávat například v týdenních vzdálenostech, nebo v čase stráveném sportem. Můžeme se také účastnit různých výzev jako třeba uběhnout 200 km za měsíc, sportovat

[1] Strava - https://www.strava.com



Obrázek 5.1: Strava jako aplikace pro záznam sportovní aktivity

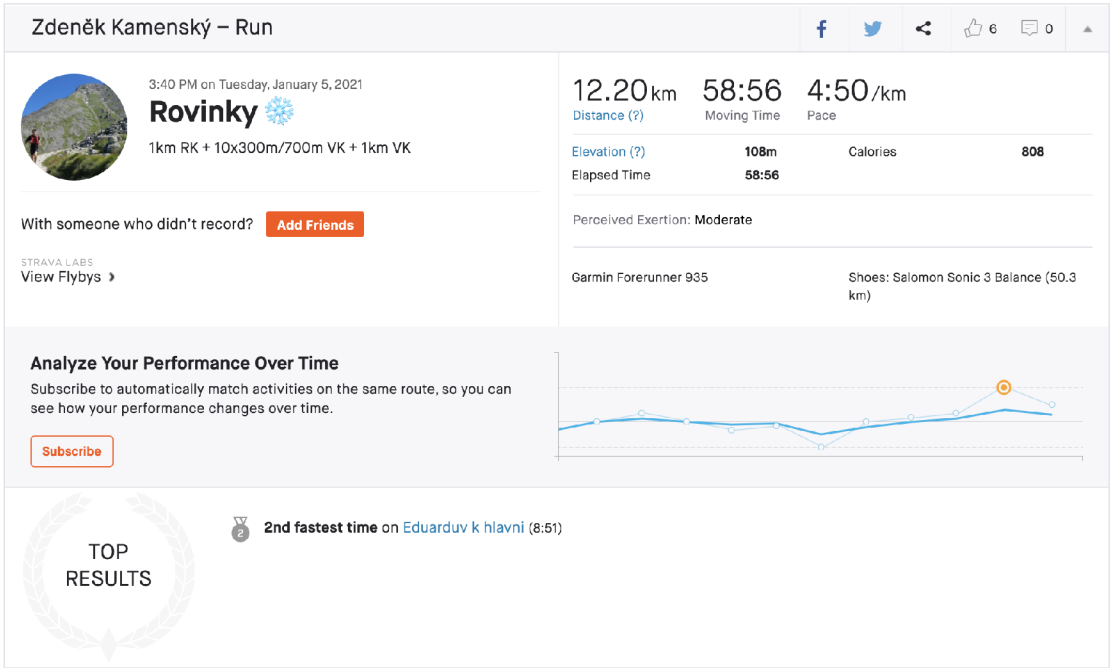


Obrázek 5.2: Část detailu aktivity v aplikaci strava

10 hodin týdně a tak podobně. Za tyto výzvy získáme nejrůznější trofeje a odznaky. Strava také zaznamenává osobní rekordy, kupříkladu nejrychlejších 50 km na kole, nejdelší uběhnutou vzdálenost, nejrychlejších uběhnutých 10 km, nejdelší uplavanou vzdálenost a jiné. Zajímavým prvkem a zpestřením jsou zde i takzvané segmenty. Jedná se o části nějaké trasy, kde se sportovci řadí do žebříčků podle časů. Stačí, aby někdo takový úsek manuálně v aplikaci vytvořil a pojmenoval, a všem uživatelům, kteří tudy poběží (pojedu na kole atd.), se ukáže, jak si vedou v porovnání s ostatními. Segmenty se často vytváří ve stoupání na nějakou horu nebo kopec, na rychlých rovinkách či technických seběžích nebo sjezdech. Předplatitelé aplikace vidí podrobné žebříčky segmentů včetně žebříčků dle pohlaví, věku nebo kalendářního roku, kdy byl segment absolvován.

Dále je také možné vidět týdenní souhrn uběhnutých, ujetých i uplavaných kilometrů a vést si inventář své sportovní výbavy. Uživatel může zadat obuv, ve které běhá, a kola, na kterých jezdí, a tím si počítat pro daný kus vybavení absolvovanou vzdálenost. Dále je také možné si ke každé aktivitě přidávat fotografie, pocity a popisky. Na detailu aktivity se

nachází nejrůznější statistiky. Vyskytuje se zde výpis jednotlivých kilometrů včetně tempa, nadmořské výšky a srdečních tepů. Jsou zde i grafy se zmiňovaným srdečním tepem, nadmořskou výškou, tempem, kadencí kroků či výkonem v průběhu celé sportovní aktivity. V neposlední řadě se u každé aktivity nachází údaj o tom, s jakým úsilím byla absolvována a jestli se jedná o obvyklé úsilí, vyšší, nebo nižší v porovnání s jinými aktivitami. Webová verze aplikace je o něco více podrobnější než její mobilní sestra, nejedná se však o veliký rozdíl. Na obrázcích **5.3** a **5.4** vidíme příklad detailu aktivity.



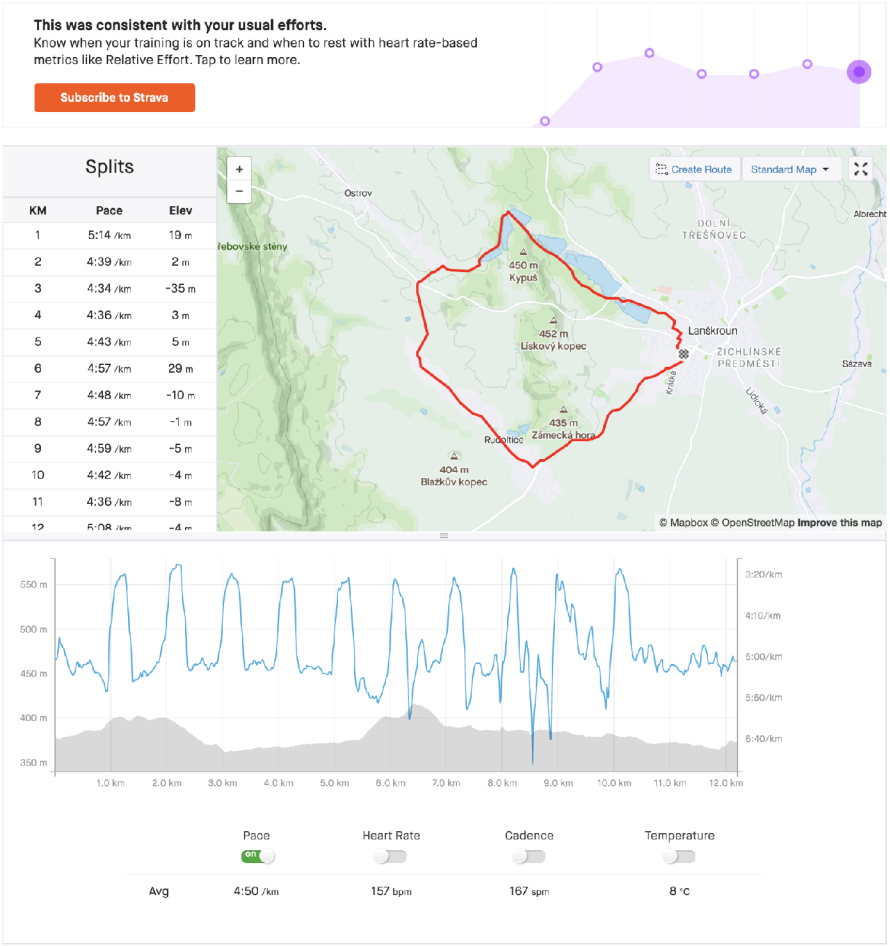
Obrázek 5.3: Detail aktivity ve webové verzi aplikace Strava 1. část.

Nevýhod má však aplikace hned několik. Tou nejmarkantnější je kompletní absence sportovních týmů, klubů a trenérů. Nelze zde mít uzavřené skupiny sportovců a trenérů. Chybí také pokročilejší nástroje pro analýzu sportovních dat, jako například automatické profilování sportovců dle jejich tréninků. Podkladů pro tvorbu dalších tréninků a sportovních cílů na základě aktuální formy je zde také velmi málo. Tím se aplikace profiluje jako pouze základní analytický nástroj pro sportovní aktivity a spíše se zaměřuje na jednotlivce a plní funkci výše zmiňované sportovní sociální sítě.

5.2 Garmin Connect[2]

Další velmi kvalitní a užitečnou aplikací využívanou ve sportovním světe ja aplikace Garmin Connect z dílny výrobce sportovních hodinek a sporttesterů Garmin. Opět se jedná o mobilní i webovou variantu. Aplikace je zdarma v celém svém rozsahu, avšak využívat ji mohou pouze uživatelé zařízení značky Garmin. Nepodporuje synchronizaci dat z jiných záznamových zařízení, ani ruční import sportovních aktivit pořízených prostřednictvím například aplikací v chytrých telefonech. Toto úzké zaměření patří mezi její hlavní nevýhody.

[2]Garmin Connect - https://connect.garmin.com



Obrázek 5.4: Detail aktivity ve webové verzi aplikace Strava 2. část.

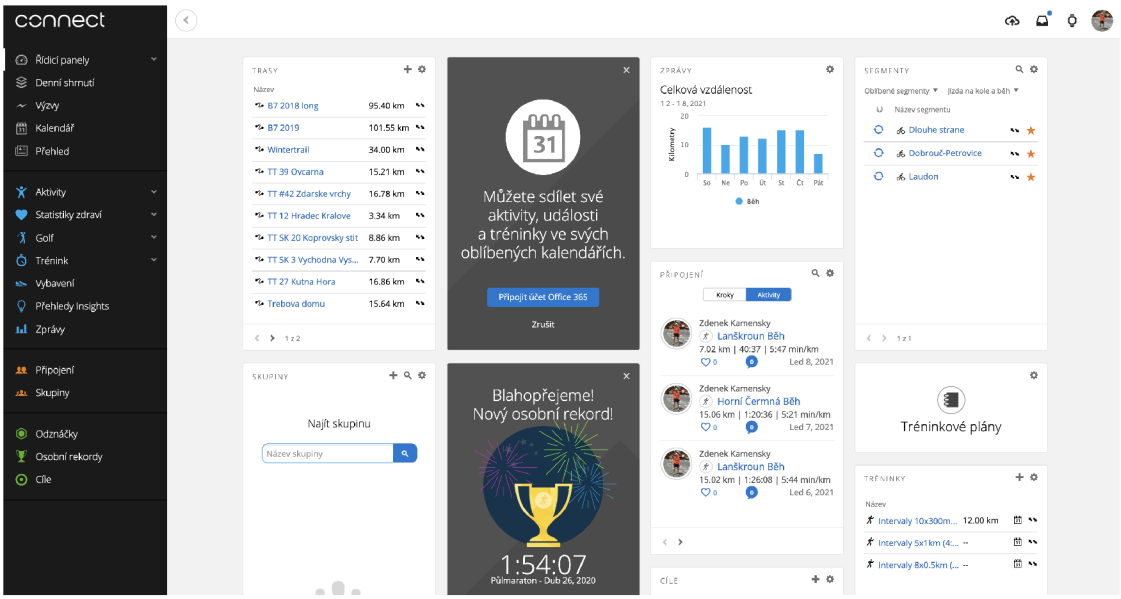
Pomineme-li tuto skutečnost, pak se jedná o velmi komplexní sportovní aplikaci s rozsáhlou řadou užitečných vlastností. Žádný jiný výrobce sportovních hodinek v současnosti nemůže nabídnou jinou tak precizně zpracovanou a rozsáhlou sportovní aplikaci.

Opět jako u Stravy (5.1) se mobilní a webová verze příliš neliší. Pouze verze pro webové prohlížeče má možná přehlednější uživatelské rozhraní a snazší ovládání. Záleží však na preferencích. Obě aplikace obsahují přehledný výpis sportovních aktivit nejrůznějších odvětví od běhu, cyklistiky přes plavání, běžecké a sjezdové lyžování až po golf a další. Na detailní stránce každé aktivity je možné vidět různě podrobné statistiky a přehledy včetně souhrnů za období od týdnů po roky. Další velkou částí aplikace jsou statistiky zdraví sportovce. Zde se nachází statistiky kroků, spánku, hydratace, srdečního tepu, hmotnosti, stresu a přijatých a vydaných kalorií. Nechybí ani každodenní souhrn těchto údajů ve spojení s absolvovanými aktivitami. Vše je vidět přehledně na hlavní stránce aplikace (dashboard).

Tím, že je aplikace vytvořena na míru uživatelům sportovních hodinek přímo této značky, může čerpat výhody z přímého propojení mezi ní a hodinkami. Uživatel má možnost živě sdílet svoji aktivitu včetně trasy, rychlosti, absolvované vzdálenosti a dalších údajů. Například při ultra dlouhých závodech či výzvách může podpůrný tým kontrolovat jeho polohu a být včas na občerstvovacích stanicích. Dále si může sportovec nahrávat do hodinek

trasy vytvořené přímo v aplikaci, nebo trasy do ní importované pomocí souborů GPX. Mezi další výhody patří tvorba tréninků na míru a tréninkových plánů, které je možné si opět nahrát do zařízení. Nechybí zde výzvy, cíle, osobní rekordy, různé odznaky a další.

Mezi nevýhody opět patří absence trenéra a sportovních týmů. Uživatel se může do jisté míry trénovat sám pomocí plánování tréninků a tréninkových plánů. Vidí také různé přehledy o absolvovaných aktivitách od srdečního tepu přes nadmořskou výšku až po tempo. Ve spoustě případů to však nestačí. Opět jako v aplikaci Strava (5.1), i v Garmin Connect chybí možnost řízených tréninkových skupin a týmů včetně trenérů. Není možné, aby zde sportovci plánovala tréninkové plány jiná osoba. Nejsou zde ani pokročilé metody analýzy sportovních dat, jako již výše zmíněné profilování sportovce dle jeho výkonů a podobně. Navíc, jak bylo zmíněno hned v úvodu této podkapitoly, je aplikace určená pouze pro uživatele zařízení značky Garmin. Na obrázku **5.5** je vidět hlavní stránka aplikace a na obrázcích **5.6** a **5.7** detail aktivity.

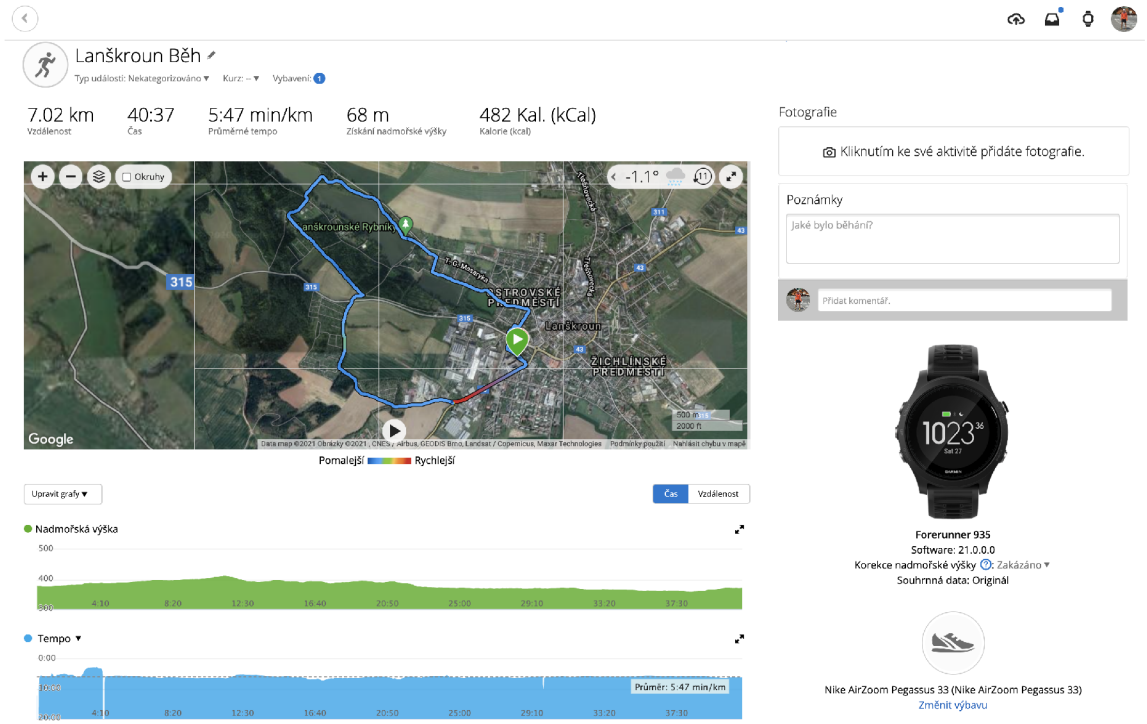


Obrázek 5.5: Přehledná hlavní stránka aplikace Garmin Connect

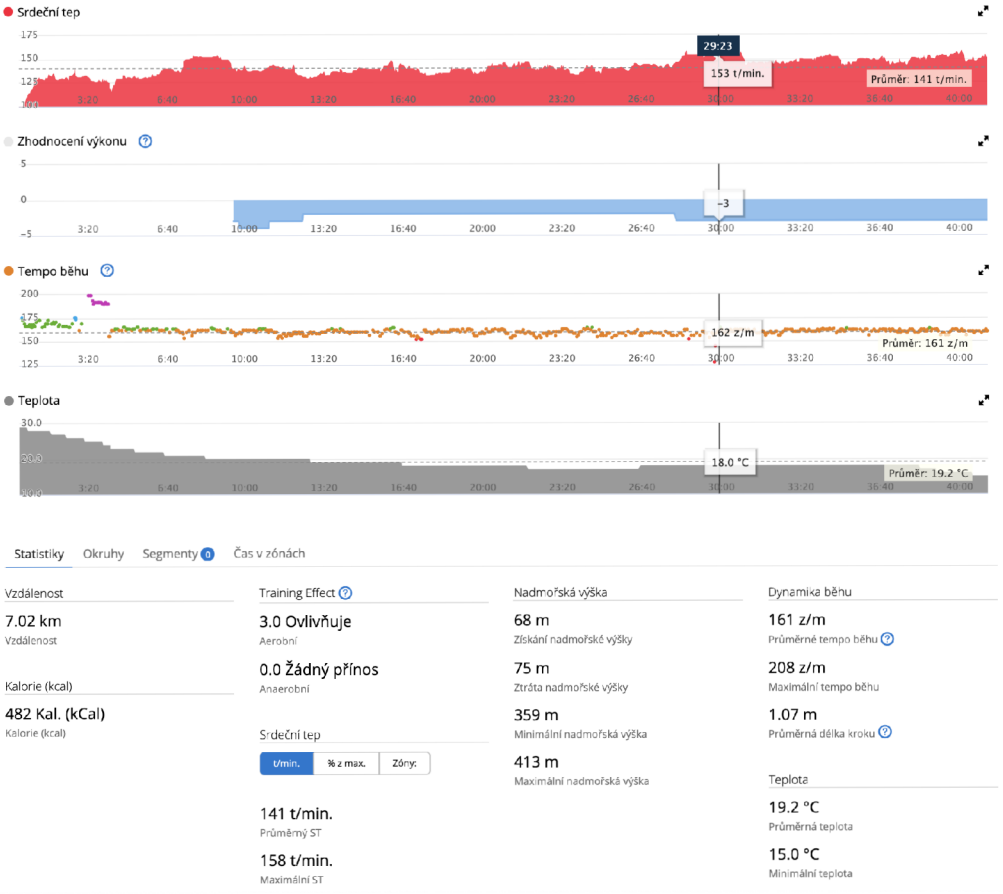
5.3 Další příklady

Spousta výrobců sportovních hodinek a sporttesterů má své vlastní aplikace, jako například Suunto Movescount[3] (Suunto App Mobile) od společnosti Suunto, nebo Polar Flow[4] od společnosti Polar. Všechny tyto aplikce jsou opět omezeny tím, že je možné je používat jen se zařízením výrobce. Existují i další určené spíše pro chytré hodinky s možným využitím pro sport (ne pro to primárně určené), jako třeba My Watch od Applu, SHealth[5] od Samsungu a další. Ani jedna z těchto aplikací však v současné době nedosahuje takové kvality a využitelnsoti pro analýzu sportovních dat a trénování, jako výše zmíněná Strava nebo Garmin Connect.

[3]Movescount - https://www.movescount.com
[4]Polar Flow - https://flow.polar.com
[5]SHealth - https://play.google.com/store/apps/details?id=com.sec.android.app.shealthhl=csgl=US



Obrázek 5.6: První část detailu aktivity v aplikaci Garmin Connect



Obrázek 5.7: Druhá část detailu aktivity v aplikaci Garmin Connect

Kapitola 6

Analýza

Jak již bylo uvedeno v přechozích kapitolách, cílem této práce je vytvořit komplexní on-line (webový) tréninkový deník. Využití by měl najít mezi úplnými hobby sportovci, pokročílejšími amatéry, výkonnostními ambiciózními atlety i lidmi, které sport živí, tedy atlety profesionály. Každá z těchto skupin má trochu jiné potřeby a požadavky, avšak všechny spojuje touha mít přehled a pořádek ve svých sportovních aktivitách. Ať už je uživatel z jakékoliv výše zmíněné skupiny sportovců, vždy rád vidí, zda jeho trénink nese kýžené ovoce, zda se zlepšuje, zhoršuje, nebo jak na tom je jeho aktuální forma. Další cílovou skupinou jsou trenéři. Ke své práci potřebují, kromě osobního kontaktu a rozpravy se sportovci, mít také k dispozici objektivní informace. Je nutné, aby trenéři měli přehled o aktuální formě svěřenců, nebo ve chvíli, kdy se celý tým připravuje na určité závody, je potřeba, aby trenér věděl, na jaký typ závodu se daný atlet hodí. Někdo má aktuálně lepší formu v kopcích, jiný naopak na rovinách.

6.1 Registrace a přihlášení

Jelikož se jedná o webový tréninkový deník dostupný odkudkoliv, kde je přítomné internetové připojení, zařízení pro prohlížení (chytrý telefon, počítač, ...) a webový prohlížeč, jsou zde umístěná data dostupná všem. Proto je nutné, aby každý uživatel měl svůj vlastní účet chráněný heslem, kde se nachází pouze jeho sportovní aktivity a statistiky. Tím si chrání svá data před tím, aby je nemohli vidět či získat jiní uživatelé systému. Ostatní uživatelé tak mohou vidět pouze ta data, u kterých to daný uživatel povolí. Případně je tato viditelnost omezena pouze na trenéra nebo tým. Nově příchozí uživatel se tedy může registrovat do systému. Zadá zde několik základních údajů jako jméno, příjmení, pohlaví, věk a další. Jednoznačný identifikátor uživatele je vzhledem k praktičnosti jeho e-mailová adresa. Ta je všeobecně unikátní a každý uživatel má svoji vlastní. Dále uživatel zadá dostatečně silné heslo a následně jej ještě potvrdí. Tímto se zaregistruje do systému. Ještě před dokončením registrace je nutné vybrat, zda je uživatel trenérem, nebo atletem a potvrdit e-mailovou adresu. Registrací je zároveň přihlášen a může dokončit svůj profil atleta, případně trenéra. V tu chvíli má uživatel před sebou takzvanou tabulu rasu. Není součástí žádného týmu a nevidí žádné svoje ani cizí aktivity. Uživatel se může do systému libovolně přihlašovat pomocí zvolených údajů (kombinace e-mailové adresy a hesla), nebo se z něj odhlašovat.

6.2 Nastavení a úprava profilu

Každý přihlášený uživatel má možnost zobrazení vlastního profilu i profilů jiných uživatelů, které může v systému vyhledat. Svůj profil může také upravovat. Má možnost měnit jméno, heslo a další údaje. V případě atleta je možné nastavit výšku, váhu, klidový srdeční tep (2.10.1), maximální srdeční tep (2.10.2), laktátový práh srdečního tepu (2.10.3) a hodnotu VO2max (2.10.4). Tyto hodnoty může atlet v průběhu času libovolně upravovat. Počítají se z nich však některé hodnoty sportovního výkonu a je nutné na to při změnách brát zřetel.

6.3 Import aktivit

Každý atlet má možnost importovat do aplikace své aktivity. Jak již bylo uvedeno v sekci 3.3, pro import záznamů sportovních aktivit byl z uvedených důvodů výbrán formát Flexible and Interoperable Data Transfer (FIT). Sportovec může propojit svůj účet přímo s účtem Garmin a automaticky synchronizovat aktivity do informačního systému. Data jsou přenášena právě ve zmiňovaném formátu. Další možností je ruční nahrání souborů tohoto formátu do systému. Ať už se jedná o první či druhý případ, systém rozloží data o daných aktivitách na jednotlivé části a vše potřebné uloží do databáze. Uživatel pak ihned vidí své nové aktivity v systému. Ty je možné samozřejmě ze systému také odstranit. V případě automatické synchronizace ze serverů Garmin tyto aktivity už nebudou znovu synchronizovány, a pokud je uživatel bude chtít mít opět v systému, musí zvolit cestu manuálního importu.

6.4 Přehled aktivit

Uživatel systému vidí přehledný seznam svých aktivit ať v dlaždicovém, tak i tabulkovém zobrazení. Aktivity jsou stránkovány a je možné mezi jednotlivými stránkami přecházet. Každá aktivita ve výpisu obsahuje několik základních informací jako třeba tempo nebo rychlost, celkovou vzdálenost, čas a další. Dále se zde nachází mapa trasy aktivity, její název a datum započetí. V tabulkovém výpisu jsou data stručnější a bez mapy trasy. Aktivity je možné řadit podle různých kritérií a filtrovat je.

6.5 Detail aktivity

Detail aktivity by měl být primárně přehledný a smysluplně uspořádaný, ale zároveň by měl obsahovat dostatečné množství informací. V žádném případě by neměl na uživatele působit chaoticky a zároveň by se nemělo stát, že bude přehlednost dosažena na úkor nějaké důležité informace. Vše by mělo být jasné a smysluplné.

V záhlaví aktivity jsou základní informace jako vzdálenost, čas, tempo, relativní úsilí, relativní náročnost trasy, relativní výkon (všechny tři viz 6.5.1) a další, podobně jako na dlaždici výpisu aktivit. Následuje mapa, na které je možné měnit zabarvení trajektorie pohybu dle rychlosti nebo tempa, srdečního tepu a nadmořské výšky. Následují grafy nadmořské výšky, srdečního tepu a rychlosti, nebo tempa. Nechybí ani graf procentuálního rozložení srdečního tepu do jednotlivých tepových zón sportovce. V další sekci detailu aktivity se nachází přehledná tabulka s jednotlivými kilometry. V tabulce jsou vypsány zajímavé základní informace daného úseku (podobně jako v záhlaví aktivity). Zároveň je každý takový kilometr označen štítkem podle toho, zda se jedná o prudké stoupání, stoupání, rovinu, zvlněný terén, klesání nebo prudké klesání. Toto je realizováno automaticky pomocí dolo-

vání dat a strojového učení, v tomto případě pomocí klasifikace. Každý kilometr uvedený v tabulce je možné rozkliknout a zobrazit modální okno s detailnějšími informacemi včetně grafů podobných těm na detailu aktivity. Zmiňovaná tabulka se nachází v jedné ze záložek. Další záložky obsahují výpis podobně náročných tras a aktivit v blízkosti. To je opět realizováno pomocí dolování dat s využitím shlukování.

6.5.1 Relativní náročnost trasy, relativní výkon a relativní úsilí

Jedná se o pokročilé metriky měření sportovního výkonu a náročnosti absolvované aktivity. Podrobný detail implementace těchto metrik bude popsán v kapitole Implementace.

Relativní náročnost trasy Relativní náročnost trasy je metrika určující, jak je daná absolvovaná trasa náročná. Bere se v potaz náročnost trasy a její profil. Čím je trasa delší, nebo obsahuje prudší a delší kopce, tím je náročnější. Naopak mírné kopce dolů trasu zlehčují. Prudké seběhy dělají však trasu opět o něco náročnější. Vše záleží na daném sklonu. Do náročnosti trasy není možné připočítat terén (skály, kameny, kořeny), jelikož tato infromace není nikde obsažena a není možné ji extrahovat ani z mapy.

Relativní výkon Vychází z náročnosti trasy, viz předchozí odstavec. Bere v potaz trasu, to jak je náročná, zda jsou jednotlivé úseky po rovině, z kopce, nebo do kopce a k tomu připočítá rychlost (tempo) sportovce v daném úseku.

Relativní úsilí Značí, jaké úsilí musel atlet nebo sportovec vynaložit pro absolvování aktivity. Někdy nemusí být na první pohled z trasy a tempa zřejmé, jak náročné to pro sportovce bylo. I on sám může cítit, že to bylo snadné. Pocity jsou však subjektivní a mohou být ovlivněny například náladou jedince a nemusí být příliš vypovídající. Metrika bere v úvahu čas a vzdálenost strávené v jednotlivých tepových zónách.

6.6 Hlavní stránka - přehled

Na hlavním přehledu může atlet vidět nejrůznější souhrny a statistiky od uběhnutých kilometrů za různá časová období přes nejdelší aktivitu, nebo aktitivu s největším úsilím, výkonem a náročností po vývoj časů strávených v jednotlivých zónách srdečního tepu.

6.7 Trenéři

Trenéři jsou tu od toho, aby trénovali své svěřence. To je jejich cíl. Aplikace by se měla co nejvíce snažit jim to zjednodušit a usnadnit jim rozhodování o nasazení sportovců do závodů nebo plánovaní dalších tréninků.

Trenér má možnost zakládat týmy a zvát do nich sportovce. Zároveň je také může z týmů odebrat. Pokud by se trenér rozhodl, že bude trénovat pouze jednoho atleta, stejně pro něho musí založit tým, aby mohl vidět jeho statistiky a záznamy o aktivitách. Trenér může jednotlivým členům svých týmů ordinovat tréniky přímo v prostředí týmu. V každém týmu může být jeden, nebo i více atletů. To samé platí o trenérech. Pordobnější popis fungování týmů je v následující sekci.

6.8 Týmy

V týmu se organizují sportovci a jejich trenéři. V každém může být libovolný počet trenérů a jejich svěřenců, avšak trenér zde musí být vždy minimálně jeden. Týmy zakládají trenéři a zvou do nich sportovce, kteří mohou pozvánku buď odmítnout, nebo přijmout a tím se stávají součástí týmu. Jednotliví sportovci vidí ostatní v týmu včetně jejich aktivit a základních statistik. Trenéři pak vidí aktivity všech svých svěřenců, přičemž u každého z nich vidí spoustu podrobných přehledů a statistik, například jaké je procentuální složení tréninků daného svěřence. Tedy jaký je jeho podíl tréninků do kopců nebo po rovinách, v jakých tepech trénuje a tak dále. Trenéři také vidí souhrnné statistiky celého týmu a vidí, kteří atleti mají nejvíce naběháno, kteří mají nejvíce naběháno ve vysokých intenzitách a který atlet je spíše vrchař, běžec do kopců, specialista na seběhy nebo rovinář. Každému z nich mohou na základě statistik plánovat další tréninky přímo prostřednictvím týmu.

6.9 Vyhodnocení rozhovorů s atlety a trenéry

Celá aplikace byla konzultována s několika hobby běžci, atlety i trenéry. Běžci všech typů a výkonnostních skupin mají rádi aplikace jako Strava (viz 5.1) a jim podobné, prostřednictvím nichž se mohou porovnávat s ostatními, mohou soutěžit na různých segmentech, sbírat komentáře a „to se mi líbí“ za své úctyhodné výkony a vidět základní statistiky aktivit. To vše je přínostné jak pro běžce jednotlivce, tak pro kolektiv. Zároveň pokud chtějí uživatelé vidět některé podrobnější statistiky, volí aplikace výrobců sprotovních hodinek jako Garmin Connect (viz 5.2) a další. Na čem se ale všichni shodnou je, že jim chybí objektivní odhady toho, kam jejich aktuální trénink vede. Zda je aktuální příprava naladí na rovinaté závody, nebo kopcovité traily. Dalším problémem jsou tréninkové skupiny a týmy trénované trenéry. Trenéři potřebují podrobné přehledy o svých svěřencích. Zejména v dnešní komunikačně vyspělé době dochází k tomu, že trenéři a jejich svěřenci se příliš často nepotkávají, nebo se neviděli nikdy a trénovaní probíhá na dálku. Kupříkladu spousta českých ultra-maratonců využívá služby trenérů z Rakouska, USA či jiných zemí. Ti potřebují podrobná data o tom, jak se sportovci aktuálním tréninkem profilují a které atlety vybrat a nominovat na nadcházející závody. Zde již aplikace jako Strava, Garmin Connect, nebo tabulky v programu Excel, nebo dokonce sdílené dokumenty nestačí. To a další aspekty a problémy by měla řešit aplikace vyvíjená v rámci této diplomové práce.

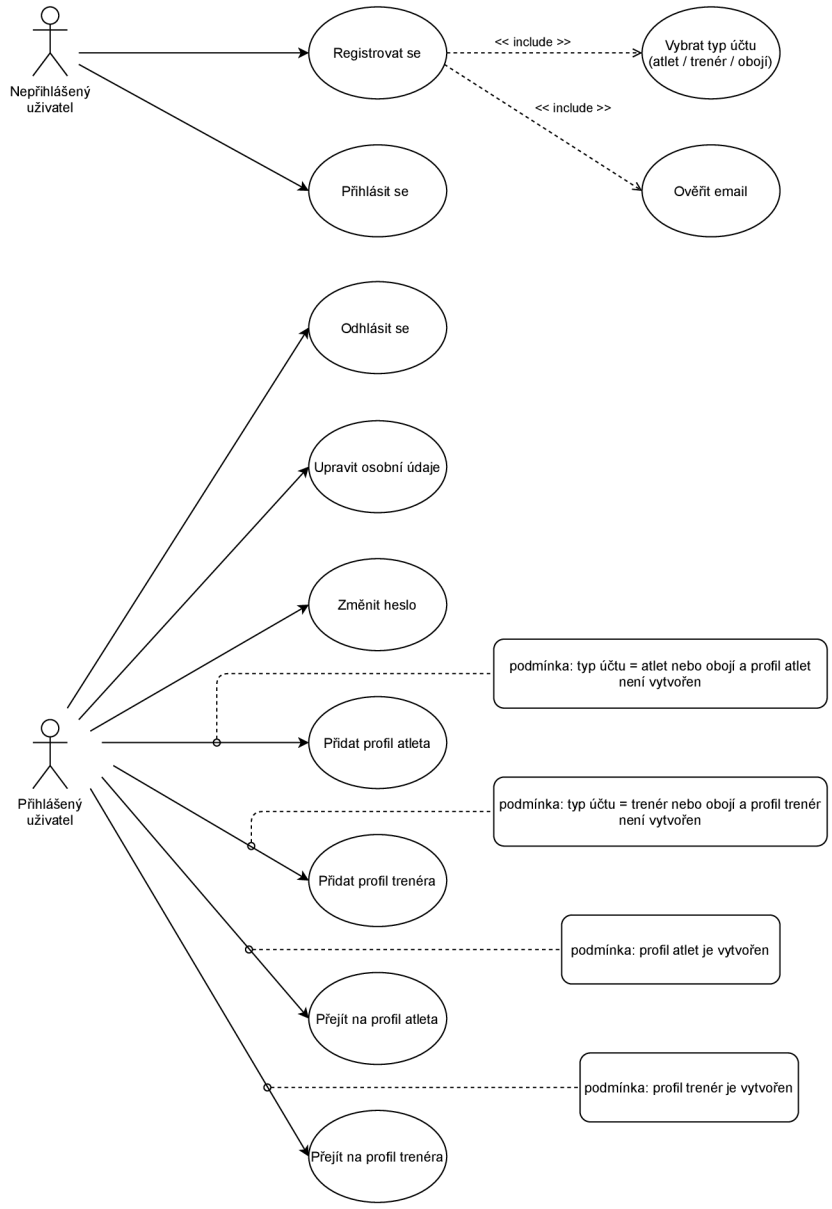
6.10 Případy použití aplikace

V této části budou stručně představeny základní případy použití aplikce/systému , které figurují jako stěžejní výstup kapitoly. Budou zde přiloženy jednotlivé diagramy a budou ve stručnosti popsány. Případy použití jsou logicky rozděleny do tří částí. První část tvoří případy pro přihlášeného a nepřihlášeného uživatele, druhá část pro atleta a třetí pro trenéra.

6.10.1 Nepřihlášený a přihlášený uživatel

Diagram pro tento případ použití se nachází na obrázku **6.1**. Nepřihlášený uživatel se může buď přihlásit, čímž se stane přihlášeným uživatelem, nebo registrovat. Při registraci je nutné vybrat typ účtu, tedy zda je uživatel atlet, nebo trenér, a potvrdit svůj e-mail. Přihlášený uživatel se naopak může odhlásit, čímž se stane nepřihlášeným uživatelem. Dále může upravit svoje osobní údaje včetně provedení změny hesla. Pokud má uživatel účet nastavený jako

trenérský a nemá doposud vytvořený profil trenéra, může tak učinit. To samé platí analogicky pro účet orientovaný pro atleta. Pokud již má takový profil vytvořený, může na něj přejít.



Obrázek 6.1: Diagram případů užití nepřihlášeného a přihlášeného uživatele

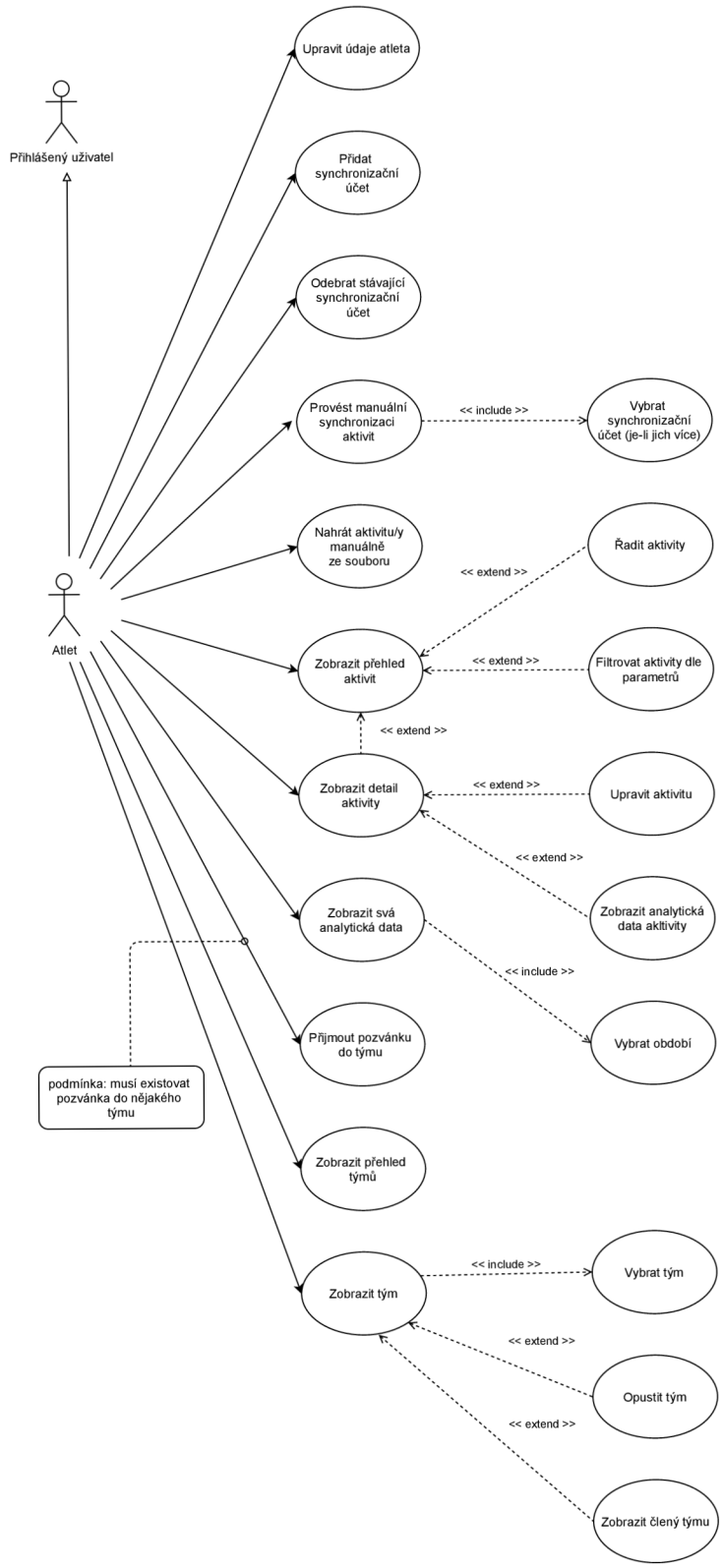
6.10.2 Atlet

Případy užití pro aktéra atlet jsou stručně a jasně znázorněny v diagramu na obrázku **6.2**, a proto zde budou uvedeny pouze některé. Atlet je speciálním případem přihlášeného uživatele, a tudíž od něj dědí i případy užití a rozšiřuje je o další. Atlet může přidávat a odebírat účet pro synchronizaci aktivit ze serverů Garmin a může data nahrávat také ručně.

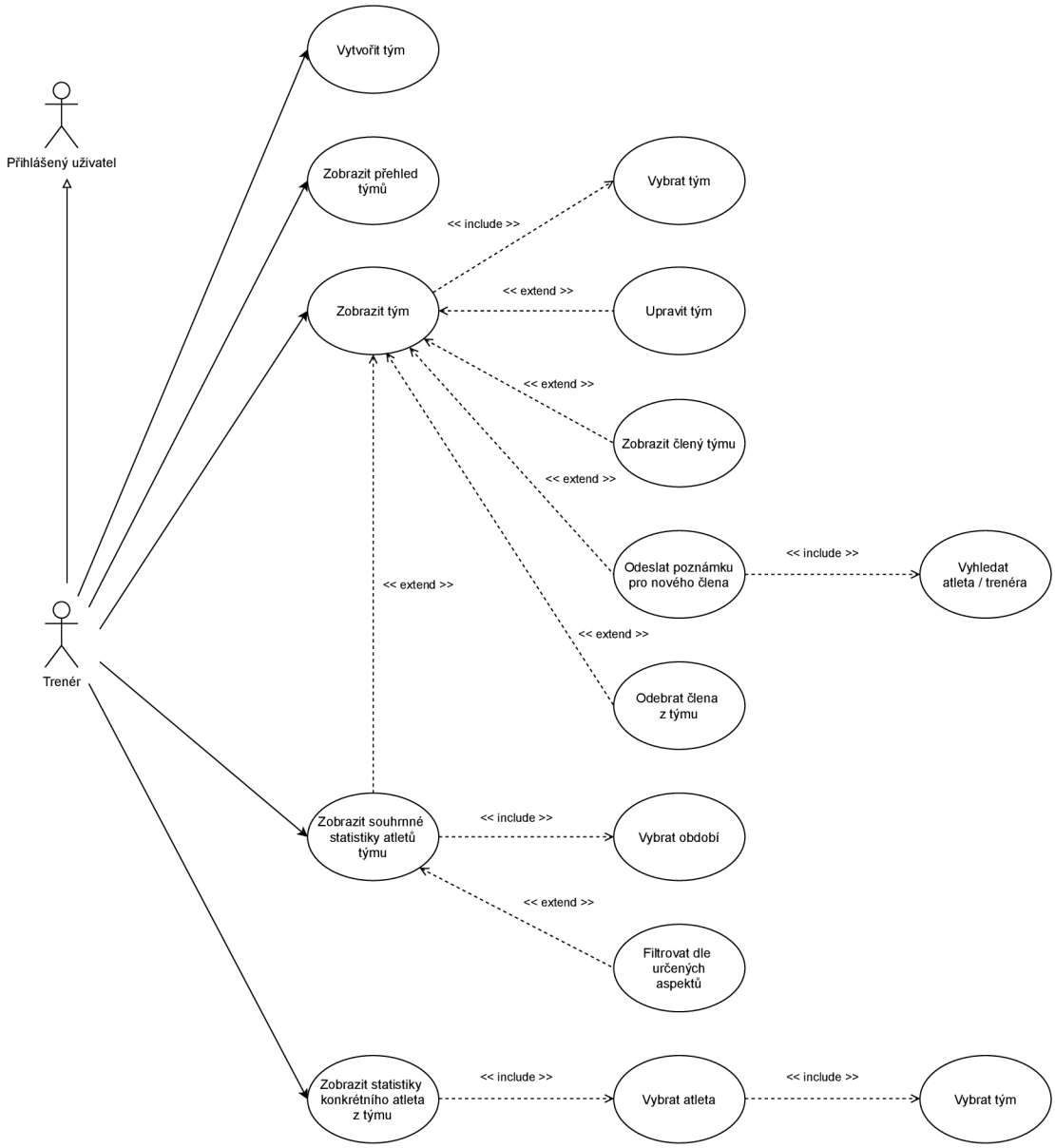
Dále si může zobrazit přehled aktivit a také detail vybrané aktivity. Další případy užití se týkají mimo jiné účasti atleta v týmu.

6.10.3 Trenér

Diagram případů užití pro aktéra trenér je vyobrazen na obrázku **6.3**. Tyto případy užití se odehrávají zejména okolo týmů a jejich organizace. Aktér trenér může vytvářet týmy a upravovat je, zvát další atlety do týmu, zobrazit stávající atlety týmu, nebo je odebrat. Dále také může zobrazovat nejrůznější souhrnné statistiky všech atletů v týmu a také přehledy pro jednoho konkrétního alteta.



Obrázek 6.2: Diagram případů užití atleta



Obrázek 6.3: Diagram případů užití trenéra

Kapitola 7

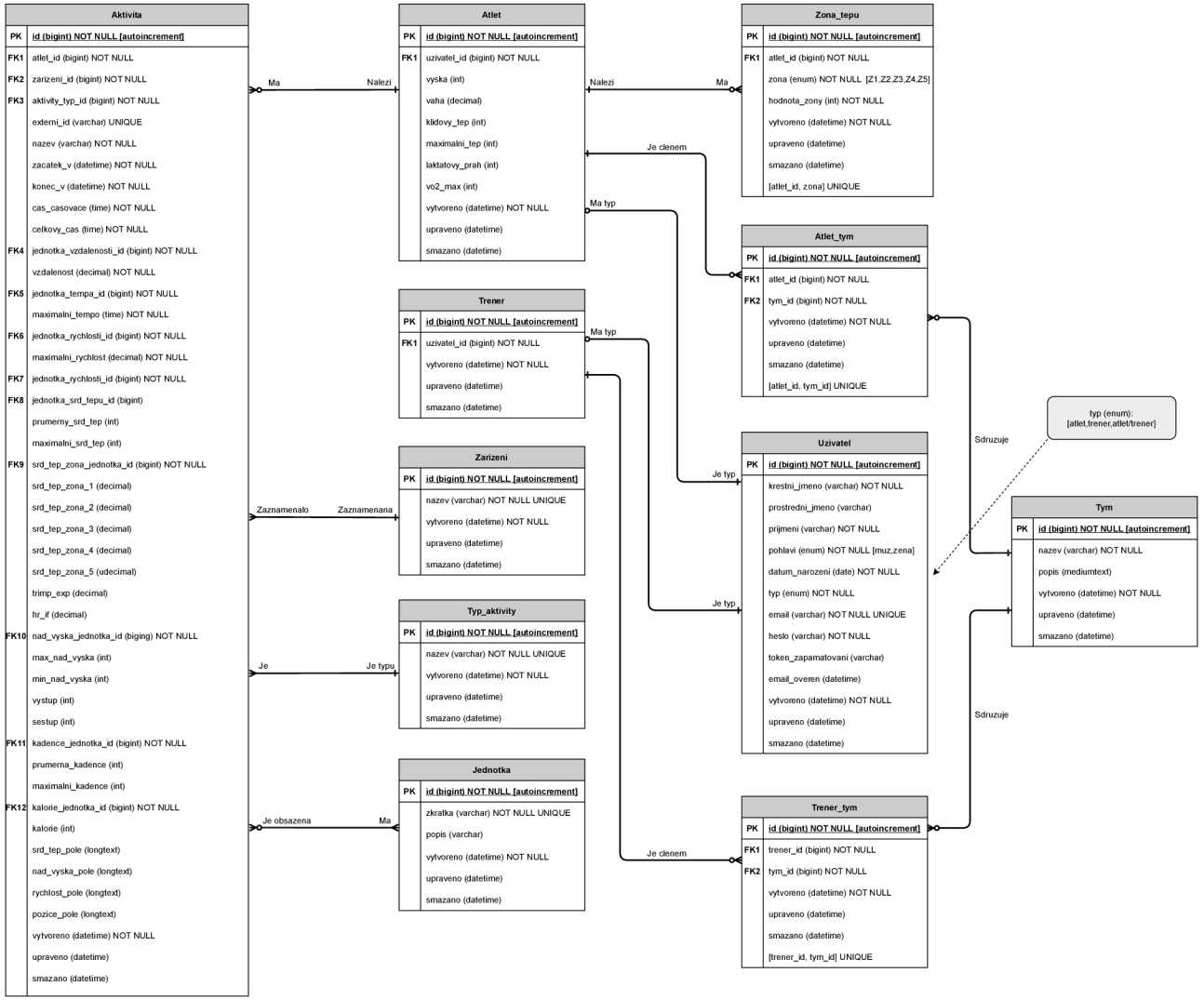
Návrh

V této kapitole bude uvedeno, jakým způsobem je řešen problém, který vyvstal z předchozí kapitoly s názvem Analýza (viz 6). Jak bylo již napsáno, nejen sportovci, ale také trenéři vyžadují systém nebo aplikaci, která jim umožní týmovou kooperaci a podporu rozhodování v plánování jak závodů, tak také tréninků. K tomu zejména trenéři mimo jiné potřebují podrobné analýzy sportovních aktivit jednolivých svěřenců a jejich porovnání v rámci týmu. Toto všechno by měl řešit komplexní on-line tréninkový deník vyvíjený v této práci. Jednotlivé části návrhu systému, respektive aplikace, budou včetně diagramů uvedeny v následujících odstavcích.

7.1 Návrh databáze

Databáze tvoří základ celého systému. Systém je na ní vystavěn a slouží mu jako perzistentní a konzistentní zdroj dat, umožňující chod systému a jako úložiště jednotlivých datových částí, které mohou definovat stav aplikace.

Při tvorbě návrhu databáze se musí brát ohled na základní prvky systému a s nimi spojené databázové entity. Jedná se tak například o entity uživatel, atlet, trenér či aktivita. Na těchto základních entitách stojí celý systém a další je pak doplňují. Dále je také nutné brát v potaz pozdější proces dolování dat a důležitý je také import sportovních aktivit do databáze, na který musí být vytvořena vhodná datová struktura. Celá tvorba ER (Entity Relationship) diagramu proběhla ve třech postupných a průběžně konzultovaných iteracích. Diagramy všech tří iterací je možné vidět v příloze B. Zde, na obrázku **7.1**, je ukázána poslední, tedy výsledná iterace. Nicméně jednotlivé iterace budou stručně popsány v následujícíh podsekcích.



Obrázek 7.1: ER diagram třetí (finální) iterace

7.1.1 První iterace

První iterace je zaměřena zejména na vytvoření entitní množiny (později tabulky) pro uchování importovaných sportovních aktivit sportovců. Nejprve bylo analýzou zjištěno, jaká data je možné získat ze souborů formátu FIT ať už manuálně nahraných, nebo automaticky synchronizovaných. Jmenovitě se jedná například o atributy pro celkovou vzdálenost, celkový čas aktivity, datum a čas začátku aktivity, maximální a minimální nadmořskou výšku, výstup a sestup atd. Speciální pozornost si zaslouží pole pro srdeční tep, nadmořskou výšku, rychlost a pozici. Jsou to serializovaná pole ve formátu JSON a nachází se v nich data o vývoji dané metriky v průběhu času či vzdálenosti. Například pro pole srdečního tepu platí, že je zde uchována hodnota srdečního tepu sportovce v průběhu aktivity a je vzorkovaná, kde vzorek je vytvořen po každých cca 20 metrech uplynulé vzdálenosti. Podobně to platí i pro ostatní pole. Dále je tato iterace zaměřena na existenci entitní množiny jednotlivých uživatelů a k nim přiřazených profilů atletů, kterým sportovní aktivity patří, respektive je abslovovali. Vše je doplněno o entitní množiny, kterými jsou množina pro zóny srdečního tepu atleta, pro zařízení, kterým byla aktivita zaznamenána, pro typ aktivity a entitní množina pro nejrůznější jednotky použitých metrik.

7.1.2 Druhá iterace

Ve druhé iteraci jsou stávající entitiní množiny z první iterace doplněny o entitní množiny reprezentující trenéry a také týmy. Dále jsou pomocí entitních množin, které řeší známý problém M:N, provázány týmy s atlety a také s trenéry. Na konci této iterace je vytvořen kompletní návrh základní struktury uspořádání dat v systému. Jsou zde uživatelé, kteří mohou být buď atlety, nebo trenéry (v unikátních případech obojími) a atleti jsou organizováni do týmů, které trénují a řídí trenéři. Každý atlet má k dispozici možnost ukládání záznamů o sportovních aktivitách, které může do systému nahrávat. Vše zmíněné je už potenciálně možné s datovou strukturou navrženou na konci této iterace.

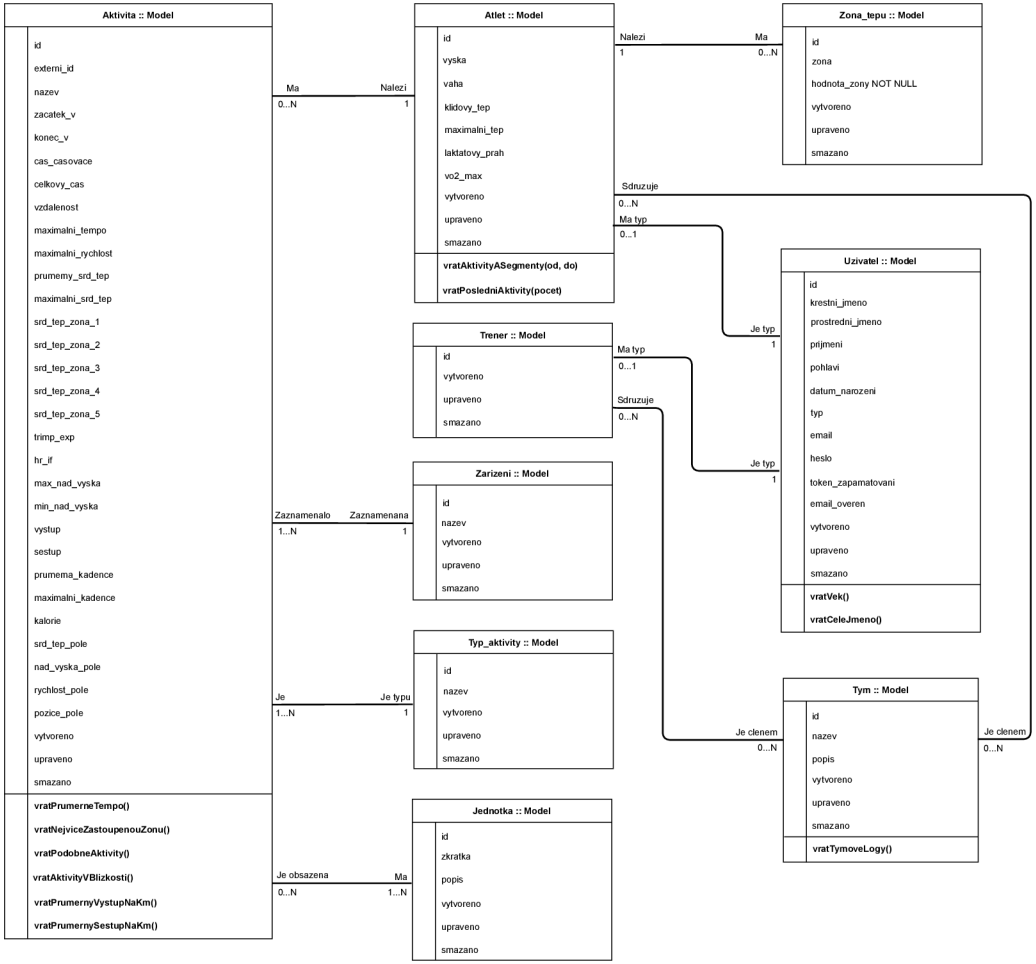
7.1.3 Třetí iterace

Z prvních dvou iterací je již vytvořen kompletní návrh struktury dat v systému. Třetí iterace se nese v duchu doplnění podrobnějších atributů do jednotlivých entitiních množin a tím se dokončuje celý návrh entit.

7.2 Návrh datových tříd

Ve chvíli, kdy je vytvořen návrh jednotlivých datových entit pomocí ER diagramu, je možné vytvořit také návrh datových tříd systému. V MVC a jiných architekturách se takovým datovým třídám říká modely. Konkrétně v prostředí Laravel byla použita třída Eloquent Model, ze které každá datová třída vychází. Jejich použití nese spoustu výhod, jako třeba oddělení datové zodpovědnosti od ostatních částí systému, kterými jsou kupříkladu řídící nebo prezentační části. Každý model má na starost danou databázovou tabulku, jejíž záznamy reprezentuje. Každý řádek či záznam načtený z databáze je takzvaně injektován do přílušného modelu a díky tomu je možné snadno a přehledně získat hodnoty různých atributů takového záznamu. Modely slouží také k samotnému získávání záznamů z databáze a mohou být prostřednictvím nich tvořeny i SQL dotazy pro vkládání záznamů a mazání.

V této práci byl návrh modelů vytvořen jako téměř identická kopie entitních množin z návrhu ER diagramu. Pouze bylo provedeno několik zjednodušení. Diagram datových tříd je možné vidět na následujícím obrázku **7.2**.



Obrázek 7.2: Diagram tříd

7.3 Návrh importu dat

Pro vyvíjený systém jsou klíčová data. A to hlavně data o sportovních aktivách jednotlivých uživatelů neboli sportovců. Ti absolvují různé tréninky, závody nebo jiné typy aktivit. Vše si přitom podrobně zaznamenávají na své sportovní hodinky nebo do sportovních aplikací chytrých telefonů. Zde se nachází podrobné informace o každé sportovní aktivitě. Zároveň je však nutné takto zaznamenaná data nějakým způsobem importovat do systému, do aplikace. Jak bylo uvedeno v kapitole 3, pro transfer sportovních dat mezi zařízeními, servery, aplikacemi a vyvíjeným systémem byl vybrán formát souborů FIT (Flexible and Interoperable Data Transfer). Do systému bude možné nahrávat tyto soubory obsahující data o aktivitách manuálně, přičemž ihned po nahrání budou data zpracována a uložena do databáze. Dalším způsobem bude přímá integrace a synchronizace s Garmin servery. Aplikace se bude umět připojit ke Garmin serverům, autorizovat daného uživatele a stáhnout požadovaná sportovní data do systému opět ve formátu FIT. Data budou zpracována a uložena stejným způsobem jako v případě manuálního importu. V této sekci bude pomocí sekvenčních diagramů modelováno, jakým způsobem probíhá komunikace s Garmin servery a jak se docílí stažení požadovaných dat. V aplikaci bude celý import implementován pomocí Service tříd k tomu určených, aby byla docílena přehlednost a oddělenost kódu a jednotlivých funkčních částí importu.

7.3.1 Autorizace a stažení seznamu aktivit uživatele

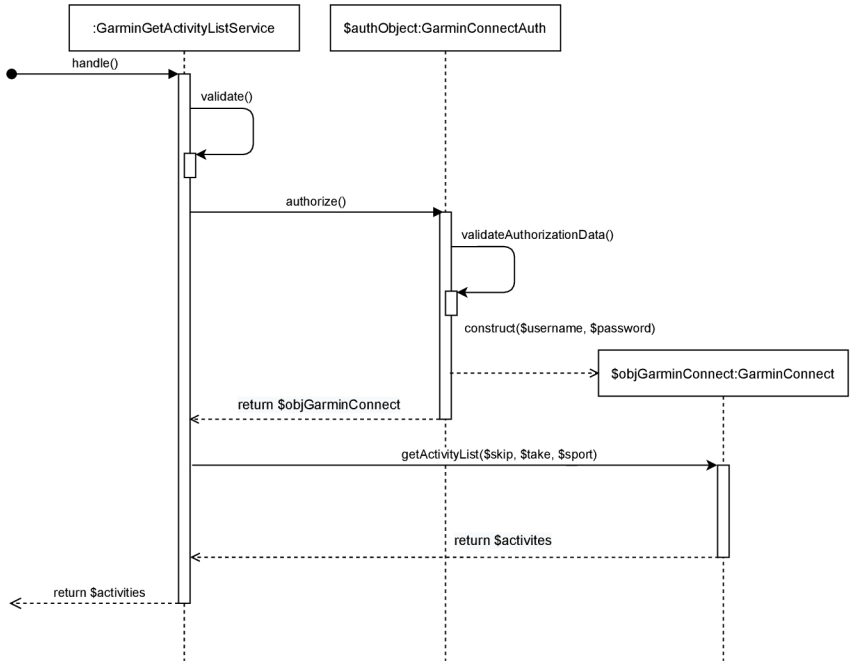
Jak je znázorněno na obrázku **7.3**, instance třídy *GarminGetActivityListService* slouží ke stažení a vrácení seznamu aktivit uživatele na místo, odkud je volána. Objekt (instance) nejprve provede validaci všech parametrů a poté zavolá objekt *GarminConnectAuth* a jeho metodu *authorize*, která opět provede validaci a následně vytvoří instanci třídy obsluhující komunikaci se servery Garmin. Při vytvoření zmíněného objektu jsou předány přihlašovací údaje uživatele a celá instance je vrácena zpět do *GarminGetActivityListService*, kde se nad ní posléze zavolá metoda pro stažení seznamu aktivit uživatele. Tento seznam je výsledkem celé sekvence kroků.

7.3.2 Autorizace a stažení záznamu dané aktivity

V předchozí podsekci bylo popsáno, jak je uživatel autorizován vůči serverům Garmin a jakým způsobem je získán seznam jeho absolvovaných aktivit. Následně je tento seznam procházen a zjistí se, jaké aktivity již byly do systému importovány a jaké ne. Neimportované aktivity jsou jedna po druhé staženy. Tento postup znázorňuje schéma na obrázku **7.4**. Stejně jako při stažení seznamu aktivit, i zde probíhá autorizace vůči serveru. Následně je nad instancí třídy obsluhující komunikaci mezi systémem a serverem (instance třídy *GarminConnect*) zavolána metoda *getDataFile*, která má jako parametry formát, ve kterém mají být sportovní data vrácena a také identifikátor požadované sportovní aktivity. Data jsou vrácena ze serveru v požadovaném formátu a navíc jsou komprimována do *.zip* archivu. V tomto kroku vše provádí Service *GetActivityAsFitFileService*.

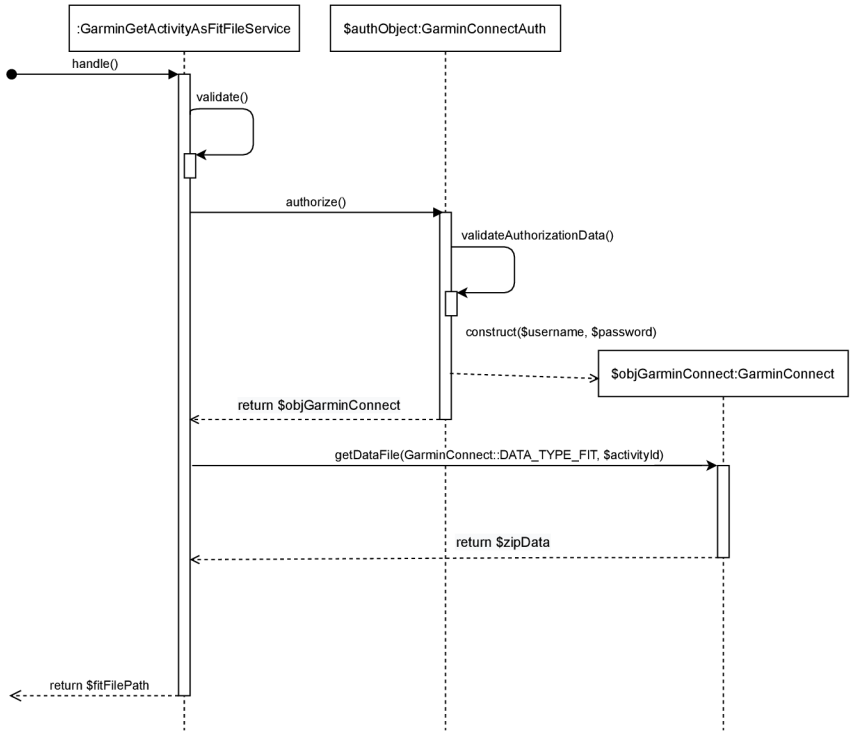
7.3.3 Celkové schéma importu sportovních dat

V předchozích dvou podsekcích bylo znázorněno a popsáno, jak to funguje uvnitř dvou klíčových Service, *GarminGetActivityListService* a *GetActivityAsFitFileService*. V tomto bodě je tedy předpokládána znalost funkčnosti obou dvou tříd, respektive jejich instancí,

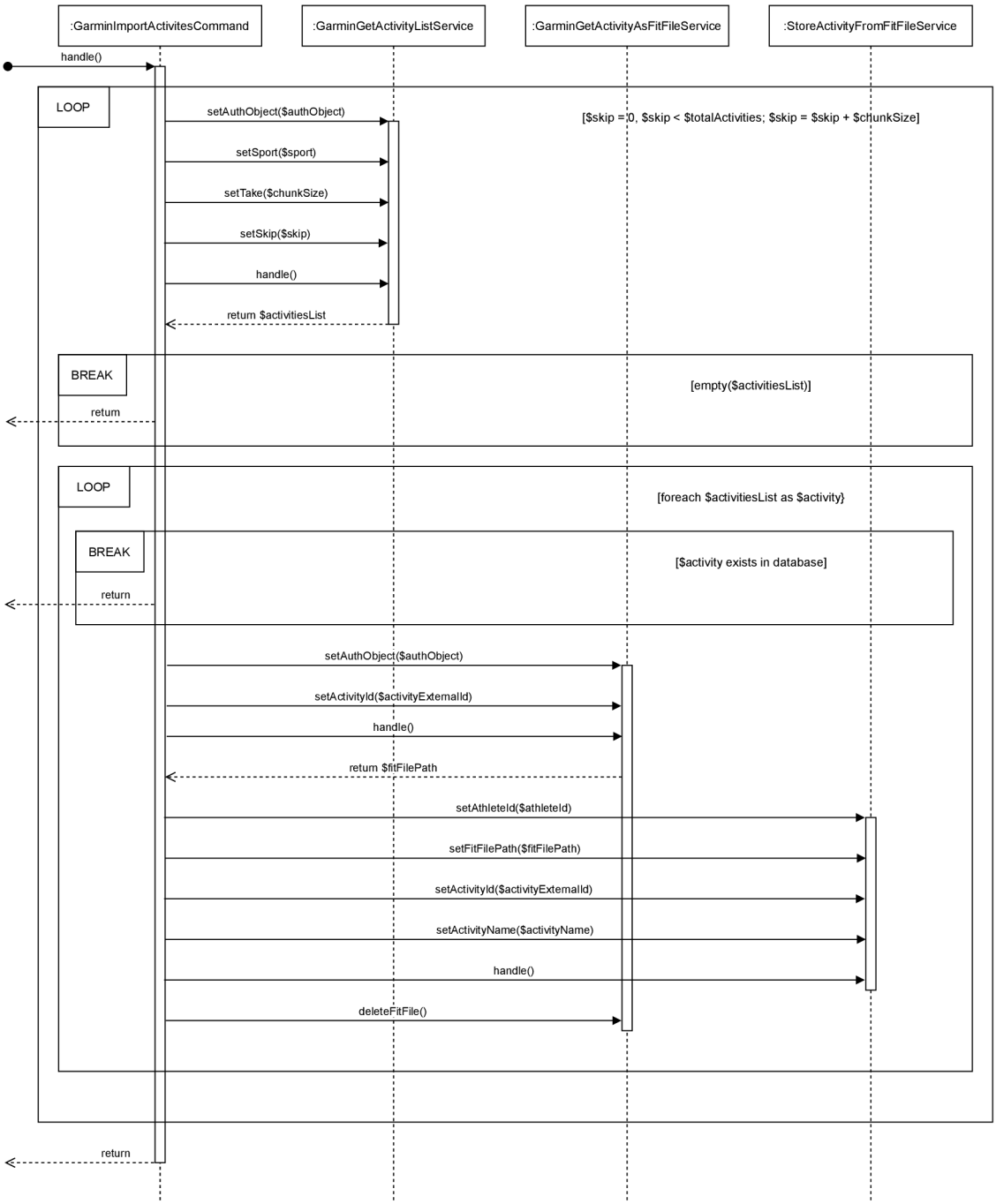


Obrázek 7.3: Diagram autorizace a stažení seznamu aktivit uživatele

které jsou pro import sportovních dat do systému využívány. Na obrázku **7.4** se nachází celkové schéma, které výše uvedené instance tříd využívá. Celý import sportovních dat má na starosti konzolový příkaz *GarminImportActivitiesCommand*. Ten po jednotlivých částech v cyklu stahuje seznamy aktivit daného uživatele. O tom pojednává první část diagramu, kde objekt *GarminImportActivitiesCommand* komunikuje s objektem *GarminGetActivityListService*. Výsledkem toho je, že příkaz má k dispozici seznam aktivit. Pokud je tento seznam prázdný, pak celý proces importu dat končí. Pokud ne, iteruje se jednotlivými aktivitami v seznamu. Pokud již daná aktivita byla nahrána do systému, opět celý proces končí. V opačném případě se daná aktivita určená jednoznačným identifikátorem stáhne do systému ve formátu FIT. O to se stará (jak již bylo vysvětleno) objekt *GetActivityAsFitFileService*. Následně je tato aktivita v daném formátu spolu s dalšími parametry předána objektu *StoreActivityFromFitFileService*, který provede kompletní zpracování detailních dat aktivity a uloží je do příslušných tabulek v databázi.



Obrázek 7.4: Diagram autorizace a stažení aktivity uživatele



Obrázek 7.5: Celkové schéma návrhu importu sportovních dat

Kapitola 8

Implementace

V několika následujících odstavcích budou nejprve uvedeny a nastíněny některé zajímavé a stěžejní technologie použité při tvorbě aplikace, přičemž následně budou objasněny a detailněji popsány klíčové postupy implementace týkající se třeba importu dat sportovních aktivit do systému, dopočítání zajímavých sportovních metrik z importovaných dat, dolování dat použitého při sestavování tréninkových profilů atletů či shlukování podobných sportovních aktivit. Krátká zmínka bude také o automatizované tvorbě databáze, způsobu vyřešení autentizace a autorizace uživatelů, o automatickém zpracování úkonů ve frontách, práci s asynchronními událostmi a v neposlední řadě o notifikacích prostřednictvím k tomu určené služby.

8.1 Použité technologie

V této sekci bude popsána část technologí, nástrojů a principů použitých při tvorbě aplikace, která je cílem a hlavním výstupem této práce. Jedná se například o framework Laravel, některé programovací jazyky, knihovnu PHP-ML pro dolování dat, služby Horizon a Pusher a další.

8.1.1 PHP[1]

Český význam zkratky PHP je hypertextový preprocesor. Původní význam znamenal Personal Home Page. Jednoduše řečeno se jedná o skriptovací programovací jazyk, který se mimo tvorbu skriptů a dalších mikroaplikací používá hlavně jako programovací jazyk serverové části webových aplikací. Je jakýmsi prostředníkem mezi datovou vrstvou a uživatelským rozhraním a většina logických úkonů a výpočtů se provádí právě v části napsané v PHP. Alternativou může být jazyk Python nebo JavaScript pro serverovou část. Nad PHP je postaven i níže popsaný framework Laravel nebo také Symfony či Nette. V této práci byl použit jazyk PHP ve verzi 7.3. kompatabilní s dalšími použitými technologiemi. [18]

8.1.2 MySQL[2]

MySQL je v dnešní době jedním z nejvíce rozšířených databázových systémů vůbec. Je postaven na relačním modelu, a tudíž je vhodný pro uchování a manipulaci s daty aplikací, jejichž základem jsou relační databáze. MySQL je šířen jako svobodný software, ovšem je

[1] PHP - https://www.php.net
[2] MySQL - https://www.mysql.com

také dostupný s mnohými vylepšeními pod různými licencemi. V současné době se často využívá v kombinaci s PHP pro tvorbu webových aplikací a systémů a je také implicitním databázovým serverem pro Laravel. Z uvedených důvodů byl vybrán i pro tuto práci. [18] [5]

8.1.3 JavaScript[3]

JavaScript je moderní a stále více využívaný objektově orientovaný skriptovací jazyk s celou řadou pokročilých vlastností, funkcí a spoustou knihoven. V souvislosti s webovými aplikacemi a systémy se používá buď pro tvorbu backend části (například v kombinaci s GraphQL[4]), nebo pro tvorbu frontend části. Ve frontend části uživatelského rozhraní funguje zjednodušeně tak, že se celý jeho kód nahraje do prohlížeče uživatele aplikace a posléze provádí dynamické úkony v závislosti na uživatelských akcích, pričemž v danou chvíli není nutné provádět interakci se serverem. V této práci byl JavaScript mimo jiné použit zejména pro komunikaci s několika API koncovými body aplikace (AJAX), pro konfiguraci a vykreslení grafů a interaktivních map a také pro oživení některých statických částí stránek. JavaScript byl využit v klasické podobě Vanilla JS[5], nicméně byla také okrajově použita knihovna jQuery[6]. [3]

8.1.4 Laravel[7]

Cílem práce bylo vytvořit webovou aplikaci či webový informační systém v podobě komplexního on-line tréninkového deníku. Jako hlavní prostředek byl vybrán framework Laravel, který se nachází na absolutní špičce nástrojů pro tvorbu moderních webových aplikací. Vyniká zejména svojí jednoduchostí a zároveň komplexností. Jedná se o rozšíření MVC architektury (Model - View - Controller), přičemž využívá mnoho dalších návrhových vzorů či technik pro návrh a implemantaci software jako třeba SOLID, DRY, KISS, Facade a další. Pro datovou a relační vrstvu postavenou primárně nad MySQL využívá ORM Eloquent a Query Builder. Pro samotnou tvorbu databáze a tvorbu demonstračních dat se používají Migrations, Seeds a Factories. V řídící vrstvě jsou hlavními články takzvané Controllers, ale také Middleware a další. Pro zobrazení výstupů se využívají Blade šablony v kombinaci s JavaScriptem a nadstavbou kaskádových stylů CSS, nazvanou Sass. Pro správu závislostí a balíčků Laravel projektů slouží níže popsaný Composer. V této práci byl použit Laravel verze 8.x. Více informací je možné dohledat v dokumentaci v online podobě na webových stránkách a v knize Laravel Up & Running (viz [16]).

8.1.5 Blade šablony[8]

Blade šablony jsou nedílnou součástí aplikací využívajících framework Laravel. Jedná se o nadstavbu či rozšíření klasického HTML o různé direktivy. Je možné využívat ty již předpřipravené, nebo si vytvořit vlastní. Dostupné direktivy jsou například pro větvení částí HTML kódu, iterování, vkládání parciálních šablon, podmiňování atd. Některé příklady je možné vidět na následujícím obrázku (**8.1**), kde se nachází větvení pomocí *@if/@elseif/@else/@endif*

[3] JavaScript - https://www.JavaScript.com
[4] GraphQL = https://graphql.org
[5] Vanilla JS - http://vanilla-js.com
[6] jQuery - https://jquery.com
[7] Laravel - https://laravel.com
[8] Blade - https://laravel.com/docs/8.x/blade

direktiv, rozšíření nadřazené šablony (*@extends*), vložení kódu do připravené sekce (*@section/@endsection*) a vložení komponenty prostřednictvím *@component/@endcomponent*. Mimo předchystané direktivy je možné definovat vlastní. V projektu bylo zavedeno několik vlastních direktiv zejména pro autorizaci a řízení uživatelských práv a rolí. Na obrázku je možné vidět direktivu @me/@endme určující, zda zobrazený uživatelský profil náleží přihlášenému uživateli.

```
@extends(Xkamen06MyTrainingConfigNamespace . '::layout.app')

@section('body')
    @me($user->id)
        @include(Xkamen06MyTrainingConfigNamespace . '::user.profile.editable')
    @else
        <div class="row">
            <h2 class="col-md-6">
                {{ '@' . $user->first_name . ' ' . $user->surname }}
                @if($user->type === 'athlete')
                    <span class="badge badge-secondary"...>
                @elseif($user->type === 'coach')
                    <span class="badge badge-primary"...>
                @else
                    <span class="badge badge-danger"...>
                @endif
            </h2>
            @if($athlete = $user->athlete)
                @sameTeamMember($athlete->id)
                    <div class="col-md-6 text-right"...>
                @endsameTeamMember
            @endif
        </div>
        @component(Xkamen06MyTrainingConfigNamespace . '::user.component.user-info-card', [
                'user' => $user,
            ])
        @endcomponent
    @endme
@endsection
```

Obrázek 8.1: Ukázka části Blade šablony

8.1.6 Bootstrap[9]

Pro základní rozložení, stylování a vzhled vyvíjené aplikace byl použit Bootstrap ve verzi 4. Vybrán byl pro svoji jednoduchost, srozumitelnost, komplexnost a výborně propracovanou dokumentaci. Samozřejmostí je použitelnost na téměř všech dostupných platformách a webových prohlížečích. Bootstrap je volně dostupná (open source) sada nástrojů kaskádových stylů (CSS) a pokročilejších komponent. Někdy bývá označován jako CSS framework. Mezi výhody patří responzivní rozložení zvané mřížka (mobile first), pokročilé JavaScript moduly a již zmíněné komponenty. Jako alternativy byly uvažovány knihovny Tailwind CSS[10] a Bulma[11], avšak nakonec byl vybrán právě Boostrap. Další stylování proběhlo také za využití jazyka Sass[12]. [15]

[9] Bootstrap - https://getbootstrap.com
[10] Tailwind CSS - https://tailwindcss.com
[11] Bulma - https://bulma.io
[12] Sass - https://sass-lang.com

8.1.7 Livewire[13]

Laravel Livewire představuje v podstatě full-stack framework propojující velmi jednoduchou a intuitivní cestou části backed a frontend. Jedná se o open source modul či framework, který je možné nainstalovat přímo do Laravel projektu. Problémy, které obvykle vývojáři řeší při vývoji webové aplikace kombinací technologií PHP, AJAX, React[14], nebo Vue.js[15], je možné jednoduše vyřešit použitím specializované PHP třídy a rozšířené Blade šablony, tedy použitím Livewire. Příklad použití pro jednoduché počitadlo je možné vidět na obrázku **8.2**. V této práci bylo s využitím Livewire vytvořeno kompletní řazení, filtrování a vyhledávání sportovních aktivit, vyhledávání týmů a uživatelů systému a také vyhledávání potenciálních členů týmu pro následné odeslání pozvánky do týmu.

app/Http/Livewire/Counter.php

```php
class Counter extends Component
{
    public $count = 0;

    public function increment()
    {
        $this->count++;
    }

    public function render()
    {
        return view('livewire.counter');
    }
}
```

resources/views/livewire/counter.blade.php

```php
<div style="text-align: center">
    <button wire:click="increment">+</button>
    <h1>{{ $count }}</h1>
</div>
```

Obrázek 8.2: Ukázka použití Livewire pro jednoduché počitadlo

8.1.8 Pusher[16] a Toastr.js[17]

Nedílnou součástí vyvíjeného systému je i služba Pusher. Jedná se o hostované flexibilní a škálovatelné API, které je navíc poměrně snadno integrovatelné s Laravel aplikacemi. Služba

[13] Livewire - https://laravel-livewire.com
[14] React - https://reactjs.org
[15] Vue.js - https://vuejs.org
[16] Pusher - https://pusher.com
[17] Toastr.js - https://codeseven.github.io/toastr

využívá komunikační kanály (channels), na kterých umí naslouchat a zachytit vyvolání Laravel událostí (Laravel Events). V JavaScript části aplikace se vytvoří instance Pusher a také konkrétního kanálu a uvede se, jaké události má kanál zachytit a co v závislosti na nich provést. Pusher byl spolu s knihovnou Toastr.js použit pro implementaci oznámení informujících uživatele například o úspěšném importu aktivit do systému. Toastr.js je knihovna v jazyce JavaScript pro neblokující oznámení typu Gnome/Growl.

8.1.9 Laravel Horizon[18]

Pro práci s frontami byl použit Laravel Horizon. Fronty jsou v této práci využívány jak pro import sportovních aktivit atletů a následné dolování sportovních dat, tak například také pro odesílání e-mailů. Některé úkony jako třeba import několika posledních aktivit a následné vypočítání tréninkového profilu atleta může trvat delší dobu. Proto je vždy vytvořen úkol (Job) a následně je vložen do určité fronty (dle typu). Každou frontu obsluhuje takzvaný worker a vykonává jednotlivé úkoly v ní obsažené. Toto všechno zaštiťuje Laravel Horizon a navíc poskytuje přehledný dashboard s informacemi o stavu front, úkolů, jejichž vykonání bylo úspěšné a tak dále.

8.1.10 Modul pro zpracování souborů typu FIT

Jak již bylo uvedeno v jedné z předchozích kapitol, FIT (Flexible and Interoperable Data Transfer) je binární formát pro přenos nejen sportovních dat. Vzhledem k tomu, že se jedná o binární formát, jeho rozložení na jednotlivé části a další zpracování není úplně intuitivní záležitostí, jako třeba zpracování XML souborů. Pro tyto účely byl využit již existující a velmi dobře fungující modul *adriangibbons/php-fit-file-analysis*[19].

8.1.11 PHP-ML[20]

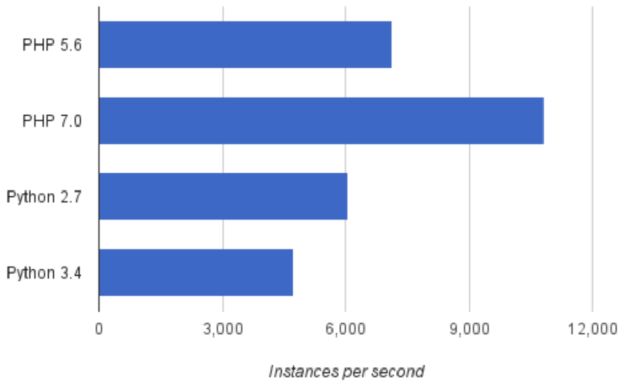
Jedním ze stěžejních bodů práce bylo dolování dat a jeho využití pro získání znalostí v oblasti sportu, potažmo sportovního tréninku a běhu. Pro tyto účely byla použita knihovna PHP-ML, která obsahuje velké množství algoritmů pro strojové učení (ML - machine learning) a tedy i pro dolování dat. V knihovně jsou implementovány například metody pro asociační pravidla, shlukovou analýzu, klasifikaci a regresi. Jak název napovídá, knihovna je určena pro programovací jazyk PHP a z tohoto důvodu a z důvodu solidní dokumentace byla použita právě pro tuto práci. Byly využity shlukovací metody DBSCAN a K-Means a klasifikační metoda NaiveBayes.

Vhodnost jazyka PHP pro dolování dat

Nejčastějšími jazyky pro dolování dat jsou jazyk R, Python nebo také Java, přičemž s PHP jako jazykem a nástrojem pro dolování dat se člověk zase tak často nesetká. Z toho vyvstává otázka, zda je pro to PHP vůbec vhodný. Od verze 7 prošlo jádro PHP kompletní přestavbou za účelem lepší strukturovanosti a celkového zrychlení všech výpočtů. Bylo provedeno také několik výkonnostních srovnávacích testů programovacích jazyků s ohledem na strojové učení. Java si samozřejmě oproti dynamicky typovaným jazykům vedla o poznání lépe, hlavně díky své nízkoúrovňovosti. Zajímavé je srovnání třeba s jazykem Python. Jak je

[18] Laravel Horizon - https://laravel.com/docs/8.x/horizon
[19] adriangibbons/php-fit-file-analysis - https://github.com/adriangibbons/php-fit-file-analysis
[20] PHP-ML - https://php-ml.readthedocs.io/en/latest

možné vidět na obrázku níže, PHP verze 7 si vede velmi dobře. Více detailních informací je možné najít v citovaném článku. Autor článku na konci uvádí, že i přes rychlost není PHP v praxi moc použitelné kvůli absenci vhodných knihoven pro strojové učení a dolování dat. Nicméně, pro tuto práci plně dostačuje výše zmiňovaná knihovna PHP-ML. [19]



Obrázek 8.3: Srovnání PHP 7.0 s PHP 5.6 a jazykem Python [19]

8.1.12 Chart.js[21]

Chart.js je obsáhlá knihovna v jazyce JavaScript. Slouží ke konfiguraci a vykreslení velkého množství různých typů interaktivních grafů. Jedná se o grafy sloupcové, spojnicové, řádkové, koláčové, plošné, bodové a další. Grafy je možné poměrně jednoduše konfigurovat a upravovat dle potřeby. V této práci byla knihovna použita pro tvorbu všech grafů, ať už se jedná o graf procentuálního podílu zón srdečního tepu, nebo třeba o graf nadmořské výšky aktivity a tempa běhu. [4]

8.1.13 Google Maps API[22]

Pro zobrazení mapových podkladů a jednotlivých tras sportovních aktivit atletů byly použity Google mapy prostřednictvím jejich API rozhraní. Mapy byly použity jak v dlaždicovém výpisu aktivit, tak na detailu každé aktivity.

8.1.14 Nástroje pro vývoj

Pro vývoj aplikace bylo použito hned několik nástrojů. PhpStorm IDE[23] pro tvorbu a editaci kódu, Git[24] a Github[25] pro jeho verzování, balíčkovací nástroj Composer[26] pro sestavení

[21] Chart.js - https://www.chartjs.org
[22] Google Maps API - https://developers.google.com/maps
[23] PhpStorm IDE - https://www.jetbrains.com/phpstorm
[24] Git - https://git-scm.com/
[25] Github - https://github.com
[26] Composer - https://getcomposer.org/

projektu a také pro připojení a správné načtení všech potřebných balíčků, knihoven a modulů a v neposlední řadě nástroj Laravel Valet[27], použitý jako vývojové prostředí vhodné pro Laravel. Server byl zakoupen od společnosti Vultr[28] a jako webový server byl nainstalován NGINX[29]. [11] [14] [2]

8.2 Obecná implementace

Základem celého on-line tréninkového deníku, který představuje webovou aplikaci, či webový informační systém, je již zmiňovaný framework Laravel ve verzi 8 a programovací jazyk PHP ve verzi 7.3. Aplikace byla vytvořena jako samostatný balíček (Laravel package), který je možné doinstalovat do jakékoliv Laravel aplikace v odpovídající kompatibilní verzi. Databáze běží nad MySQL 8 a jednotlivé databázové tabulky byly vytvořeny automatizovanou formou pomocí migrací (Migrations). Dále byla dodržena struktura architektury MVC. Modely jsou v tomto případě jednotlivé třídy typu ORM Eloquent, které jsou navíc rozšířené o třídy návrhového vzoru Presenter, který slouží k prezentaci dat z modelů v uživatelském rozhraní. Samozřejmostí je podpora serializace modelů kvůli frontám, podpora obnovitelného odstranění záznamů v databázi (soft deletes) a v neposlední řadě ukládání modelů do mezipaměti. Nad modely stojí kontrolery (Controllers), které řídí chod celé aplikace s využitím služeb (Services). Ty jsou využívány kvůli zachování větší čistoty a přehlednosti kódu a odstínění kontrolerů od přílišné logiky. Směrování URL adres na jednotlivé kontrolery řídí takzvané roury neboli routes. Mezikrokem mezi nimi jsou třídy typu middleware, které kontrolují například, zda má přihlášený uživatel ověřený e-mail a tak dále. Využíváno je také kontejnerů (Containers) pro dynamické uchování určitých dat při běhu aplikace. Pro zobrazení uživatelského rozhraní se plně využívá předností Laravel Blade šablon, jazyku JavaScript a knihoven jQuery a Bootstrap. Pro import dat do systému jsou použité nejen uvedené služby (Services), ale také konzolové příkazy (Commands) a úkoly (Jobs), které jsou vkládány do fronty. Toto byl obecný úvod do architektury aplikace a v dalších odstavcích již budou popsány konkrétní případy jednotlivých zajímavých částí implementace.

8.3 Manuální import sportovních aktivit

Z předešlých sekcí a kapitol je zřejmé, že atlet používající on-line tréninkový deník má dvě možnosti importu dat sportovních aktivit. První možností je manuální import dat. Druhou možnost představuje automatizovaný import, který však velmi úzce souvisí právě s manuálním importem a některé části z něho přímo vychází. Na několika následujících řádcích bude tedy nejprve popsán právě manuální import.

Ať už atlet pro záznam svých sportovních aktivit používá hodinky značky Garmin, Suunto, Polar, nebo jiné, ve valné většině případů má možnost své aktivity exportovat jako binární soubory dříve uvedeného formátu FIT. Stejnou možnost nabízí v dnešní době velmi populární sportovní sociální síť a aplikace v jednom Strava (detailněji popsána zde - 5.1). Takto exportované soubory pro jednotlivé aktivity je následně možné přímo nahrát do v této práci vytvořeného komplexního on-line tréninkového deníku. Soubory je možné nahrát buď po jednom, nebo po vícero naráz. Maximální počet souborů nahrávaných naráz je 20.

[27] Laravel Valet - https://laravel.com/docs/8.x/valet
[28] Vultr - https://www.vultr.com
[29] NGINX - https://www.nginx.com

Tuto hodnotu je možné konfigurovat. Soubory se nahrávají přes klasický HTML formulář a jsou uloženy do dočasného adresáře v aplikaci.

Nad nahranými soubory je následně vytvořen a spuštěn úkol (Laravel Job, spolu s Laravel Horizon popsán zde - 8.1.9), který je umístěn do fronty a čeká na obsluhu. Proces běžící na serveru (worker) poté vyjme úkoly z fronty a provádí obsluhu. Celý proces manuálního importu je tedy neblokující a uživatel má možnost provádět různé úkony, zatímco se mu do systému nahrávají aktivity. Uživatel je také upozorněn při započetí importu, při úspěšném či, neúspěšném nahrání konkrétní aktivity a po samotném dokončení importu. Samožřejmostí je validace vstupních dat.

Obsah každého souboru je nahrán do mezipaměti a následně extrahován, analyzován a rozčleněn s využitím balíčku *adriangibbons/php-fit-file-analysis* (viz 8.1.10). Jako první je získána časová značka začátku sportovní aktivity a její externí identifikátor. Dále je ověřeno, zda daná aktivita již figuruje v systému a zda není označena jako smazaná. Pokud se již v systému nachází a je označena jako smazaná, pak se obnoví (soft delete), pokud je však stále aktivní, je import této aktivity přeskočen. V opačném případě jsou jednotlivé položky, vlastnosti a metriky aktivity v případě potřeby upraveny a dopočítány (dopočítání zajímavých metrik je uvedeno dále) a vloženy do objektu. Zde probíhají drobné úpravy dat, jako například dopočítání celkového výstupu, sestupu a dalších atributů. Pro každou metrickou jednotku použitou v dané aktivitě, jako třeba tzm (tepy za minutu), min/km (minuty na kilometr), kcal (kilokalorie) a tak dále, je v případě neexistence vytvořen záznam v databázi v příslušné tabulce, a tento záznam je na danou aktivitu navázán.

Obdobným způsobem je postupováno v případě typu aktivity (klasický běh, trail běh po stezce, běh na běžeckém páse, ...) a modelu záznamového zařízení (Garmin Forerunner 945, Suunto Ambit3 Peak, ...). Zmíněný objekt je po dokončení výpočtu všech potřebných metrik a vlastností uložen v rámci jednoho záznamu do databáze. Mezi uloženými metrikami jsou například datum a čas začátku, vzdálenost, uplynulý čas, celkový čas, minimální a maximální srdeční tep, maximální rychlost, minimální a maximální nadmořská výška, celkový výstup a sestup, kalorie a další. Zároveň jsou zde uložená tři serializovaná pole. Jedná se o pole vývoje srdečního tepu v čase aktivity, pole vývoje nadmořské výšky a také rychlosti. Aktivita je prostřednictvím identifikátoru přímo vázaná ke konkrétnímu atletovi. Všechny výše uvedené kroky provádí daný Job prostřednictvím instance třídy služby (Service) s názvem *StoreActivityFromFitFileService*.

Dalším krokem importu je vzorkování aktitivy do jednotlivých segmentů dlouhých přesně jeden kilometr. Pro každý segment jsou výše uvedeným způsobem extrahovány, dopočítány a zapsány všechny metriky, jako by se jednalo o celou aktivitu. K tomu slouží opět pro tento účel implementovaná služba *StoreActivitySegmentsFromActivityService*. Jak tato služba, tak služba pro uložení celé aktivity byla vyvinuta přímo na míru vyvíjeného systému.

V úplnén závěru se ještě vytvoří a umístí do fronty několik úkolů (Laravel Jobs) využívajících dolování dat pro získání zajímavých informací týkajících se dané importované aktivity, více aktivit ve společném kontextu, nebo tréninkového profilu daného atleta. O těchto úkolech a samotné implementaci dolování dat je psáno dále.

8.4 Automatizovaný import sportovních aktivit

Pro uživatele sportovních hodinek, sporttesterů a jiných zařízení značky Garmin je připraven a implementován automatizovaný import sportovních aktivit. Tito uživatelé mají vytvořený účet u společnosti Garmin a po absolvování tréninků si svoje aktivity nahrávají na servery buď prostřednictvím Wi-Fi připojení, nebo s využitím chytrých telefonů a tech-

nologie Bluetooth. Pokud má atlet své aktivity tímto, nebo jiným způsobem nahrány na servery společnosti Garmin, může si v on-line tréninkovém deníku nastavit automatizovaný import. V uživatelské sekci je potřeba zadat Garmin přihlašovací údaje, kterými jsou e-mail a heslo, nebo login a heslo. Při zadání těchto údajů systém ověří možnost připojení na servery, tedy jestli uživatel zadal správné údaje. Pokud se připojení podaří, je import nastavený a je možné pomocí tlačítka v horní navigační liště aplikace spouštět na pozadí automatizovaný import. Nahrává se vždy posledních dvacet aktivit (opět konfigurovatelné) s tím, že pokud nějaká aktivita byla ze systému odstraněna, již se znovu nenahraje a je třeba využít manuálního importu.

Po kliknutí na tlačítko importu aktivit se vytvoří Job a zavede se opět do fronty. To však za předpokladu, že má uživatel vyplněny přihlašovací údaje, viz předchozí odstavec. Tyto údaje jsou uloženy jako záznam v databázi v příslušné tabulce uživatele. Pro zjednodušení nebyl tento fakt uveden v ER diagramu. Bezprostředně poté, co je Job vyjmut z fronty čekajících úkolů, je vytvořen autorizační objekt pro připojení instance aplikace právě přihlášeného uživatele k serverům Garmin. Třída objektu se nazývá *GarminConnectAuth*. Pokud se autorizace nezdaří, je vyvolána výjimka a celý úkon je následně ukončen. V opačném případě je nad rozhraním Garmin zavolána metoda *getActivityList* s pro daného atleta. Tato metoda vrací seznam aktivit atleta s krátkým rozšiřujícím popisem. Jako parametr je předán autorizační objekt a počet aktivit. Jak již bylo uvedeno výše, ve výchozím nastavení to je dvacet aktivit. Provolání metody s danými parametry a následné navrácení seznamu aktivit zajišťuje instance třídy služby s názvem *GarminGetActivityList*.

Zmíněný seznam aktivit je navrácen do objektu typu Job, kde je tímto seznamem iterováno a pro každý záznam je zjištěno, zda se daná aktivita již v databázi nachází (včetně označených jako smazané) či nikoliv. Aktivity v databázi již existující jsou ignorovány. Pokud se jedná o zcela novou aktivitu, je autorizační objekt a externí identifikátor aktivity předán další službě. Tentokrát se jedná o instanci třídy *GarminGetActivityAsFitFile*. Zde se nad připojením k serverům společnosti Garmin provolá metoda *getDataFile*, kde prvním parametrem je požadovaný formát souboru, v tomto případě FIT, a druhým parametrem je identifikátor aktivity. Metoda vrací komprimovaná data ve formátu ZIP, obsahující soubor v požadovaném formátu. Následně je ZIP extrahován a samotný FIT soubor uložen jako dočasný soubor do připraveného adresáře. Cesta k souboru je opět vrácena do objektu Job. Zde již následuje stejný postup krok po kroku, jak tomu bylo při manuálním importu dat. Soubor je tedy otevřen a zformátován do podoby asociativních polí, některé metriky jsou dopočítány, upraveny a je proveden zápis aktivity do příslušné tabulky v databázi. Příklad kódu implementace služby *GarminGetActivityAsFitFile* je možné vidět na obrázku **8.4**.

Pro automatizovaný import aktivit do systému je vytvořen také Artisan příkaz (Laravel Artisan Command[30]) do příkazové řádky, kde se jako parametr zadá identifikátor atleta, a pokud daný atlet má v databázi vyplněné přihlašovací údaje pro Garmin, dojde ke stažení jeho aktivit dle výše popsaného postupu. Příklad importu z konzole je možné vidět na obrázku **8.5**.

8.5 Pokročilé sportovní metriky a předzpracování dat

V průběhu importování sportovních dat do systému je dopočítáno hned několik informací o dané aktivitě sportovce. Většina z nich je poměrně triviální, jako například zjištění maximální a minimální nadmořské výšky a celkového výstupu a sestupu z pole vývoje nadmořské

[30]Laravel Artisan - https://laravel.com/docs/8.x/artisan

výšky v průběhu aktivity. Nejinak je tomu s maximálním a minimálním srdečním tepem nebo maximální rychlostí. Zároveň však probíhá výpočet několika pokročilých metrik. Jedná se o **relativní náročnost trasy**, **relativní výkon** atleta na dané trase a **relativní úsilí** atleta za celou absolvovanou aktivitu. Tyto údaje jsou vypočítány ze vzdálenosti, pole vývoje srdečního tepu, pole vývoje rychlosti a pole vývoje nadmořské výšky. Spolu s ostatními výpočty a extrakcí položek ze stažené aktivity je možné tento krok, který je stále součástí importu, považovat za fázi předzpracování dat před zahájením dolování dat. V následujících odstavcích bude detailněji popsán výpočet každé ze tří výše uvedených metrik.

8.5.1 Relativní náročnost trasy

Relativní náročnočnost trasy je získána tak, že celá aktivita je rozdělena do velkého množství úseků. Samotná četnost úseků závisí stejně jako délka každého úseku na frekvenci vytváření GPS záznamů záznamovým zařízením. Zjednodušeně řečeno, pokud dané zařízení sbírá data o poloze jednou za 10 vteřin, bude při stejné rychlosti běhu vytvořeno méně delších záznamů. Při sběru dat každé 4 vteřiny budou úseky kratší a bude jich více. Toto většinou závisí na nastavení záznamového zařízení, nebo na kvalitě GPS snímače. Při praktickém zkoumání bylo zjištěno, že velikost úseků je obvykle v intervalu 0 až 30 metrů.

Pro každý takový úsek je zjištěna jeho vzdálenost a zisk nebo ztráta výškových metrů oproti předchozímu úseku (ve vzorci označeno jako vyskovyRozdil). Z těchto hodnot je vypočítán sklon v procentech. Vzorec pro výpočet sklonu je následující:

$$sklon = \frac{vyskovy\_rozdil}{vzdalenost} * 100 \tag{8.1}$$

Další výpočet závisí na aktuálním výškovém rozdílu. Pokud je výškový rozdíl roven nule, je relativní náročnost daného úseku dána součinem jeho vzdálenosti a kvocientem nadmořské výšky, který je dán v rozmezí 1 - 4 z závislosti na nadmořské výšce. Kupříkladu úsek v nadmořské výšce do 1500 metrů nad mořem má kvocient s hodnotou 1, v nadmořské výšce 1500 až 2000 metrů kvocient 1,2 a tak dále. Vzorec pro rovinatý úsek je tedy následující:

$$relativni\_narocnost = vzdalenost\_useku * kvocient\_nadm\_vysky \tag{8.2}$$

Pro výškový rozdíl větší jak nula, tedy pro úsek do kopce, se jeho relativní náročnost počítá následovně:

$$relativni\_narocnost = (\frac{sklon + 2}{2}) * vzdalenost * kvocient\_nadm\_vysky \tag{8.3}$$

Dělitel dvě je zde použit pro zmírnění vlivu sklonu na náročnost trasy. Hodnota plus 2 se zde nachází proto, že například sklon s hodnotou 1 by daný úsek zjednodušoval i přesto, že se jedná o mírný kopec. Pro výškový rozdíl menší jak nula, který značí, že je daný úsek z kopce, popisuje výpočet tento vzorec:

$$relativni\_narocnost = (\frac{2}{sklon + 2}) * vzdalenost * kvocient\_nadm\_vysky \tag{8.4}$$

Jedná se z hlediska sklonu o opačný vzorec, než pro úsek do kopce. Klesání v určitém sklonu tedy snižuje relativní náročnost úseku stejnou měrou, jakou ji stoupání ve stejném, avšak opačném sklonu zvyšuje.

Relativní náročnost trasy je poté suma relativních náročností všech jednotlivých úseků trasy. Slovo relativní zde figuruje proto, že současné technologie nejsou schopny reflektovat ani nijak zaznamenat terén absolvované trasy. Existuje mnoho aspektů, které mohou zvyšovat, nebo snižovat náročnost prostředí, ve kterém atlet trénuje. Pro příklad uvažujme dvě trasy. Obě dvě mají vzdálenost 1 kilometr, přičemž se jedná o rovnoměrné stoupání, kde na obou trasách je nastoupáno 50 metrů. Jedna trasa vede v horském masivu po ostrých a kluzkých kamenech a místy se nachází sníh nebo led. Naopak druhá trasa vede po stěrkaté pěšině. Použitý algoritmus vyhodnotí trasy jako stejně náročné, ačkoliv rozhodně nejsou. Tato metrika je tedy velmi dobrým ukazatelem, ale neplatí vždy úplně spolehlivě. V některých, řekněme extrémních případech, je nutné použít subjektivní hodnocení atleta, které může být v danou chvíli o něco více vypovídající.

Výpočet relativní náročnosti trasy zaštiťuje instance třídy služby *RelativeRouteDifficultyService*.

8.5.2 Relativní výkon

Relativní výkon udává, jak hodnotný výkon podal sportovec v závislosti na náročnosti trasy. Jak předchozí věta naznačuje, relativní výkon přímo navazuje na relativní náročnost trasy. Jedná se o obohacení předchozího vzorce o aktuální rychlost atleta v daném úseku. Celkový relativní výkon trasy je opět sumou relativních výkonů pro jednotlivé úseky. Uvažujme N počet úseků a i, jako index úseku. Výsledný vzorec má podobu:

$$relativni\_vykon = \sum_{i=1}^{N}(relativni\_narocnost\_useku_i * rychlost\_useku_i) \tag{8.5}$$

Může tedy nastat případ, kdy sportovec běží velmi pomalu do prudkého kopce, ale jeho relativní výkon je mnohem větší, než při svižném tempovém běhu po rovině či z kopce. Za výpočet relativního výkonu zodpovídá služba *RelativePerformanceService*.

8.5.3 Relativní úsilí

Relativní úsilí vyjadřuje množství úsilí vynaložené atletem pro absolvování dané aktivity. Tato metrika je zcela oproštěna od trasy, její náročnosti i rychlosti. Aktivita je opět rozdělena na úseky, avšak nyní se nejedná o úseky délky, nýbrž času. Opět záleží na frekvenci sběru dat záznamovým zařízením sportovce (chytré hodinky, sporttester a daší). Nyní však nehraje roli vzdálenost mezi jednotlivými úseky, ale uplynulý čas. Dalšími parametry vstupujícími do výpočtu relativního úsilí je srdeční tep a zóny srdečního tepu atleta. Jedná se o pět základních tepových zón dle frekvence srdečního tepu, kde zóna jedna je nejsnazší a zóna pět naopak nejnáročnejší. Pokud atlet nemá vyplněny rozsahy, potažmo intervaly svých tepových zón, je použito výchozí nastavení, které je možné vidět v následující tabulce. V ní je také možné vidět koeficienty zón použité dále při výpočtu.

Relativní úsilí atleta vynaložené v daném úseku se vypočítá jako součin délky trvání úseku (čas), koeficientu zóny aktuální hodnoty srdečního tepu a samotné aktuální hodnoty srdečního tepu. Výsledné relativní úsilí pro celou aktivitu je pak opět sumou relativních úsilí vynaložených v jednotlivých úsecích. Výpočet probíhá ve službě *RelativeEffortService* a výsledný vzorec je následující (uvažujeme N časových úseků, přičemž i značí index úseku):

Srdeční tep (tzm)	Zóna	Koeficient zóny
<114	1	1
114 - 142	2	2
143 - 161	3	3
162 - 180	4	4
>180	5	5

Tabulka 8.1: Tabulka pro koeficienty zón srdečního tepu

$$relativni\_usili = \sum_{i=1}^{N}(koeficient\_zony\_useku_i * cas\_useku_i * srdecni\_tep\_useku_i) \tag{8.6}$$

8.6 Dolování dat

Tato část kapitoly implementace bude věnována dolování dat. V několika odstavcích bude popsáno, k jakému účelu bylo dolování dat použito, jaké infomace byly díky tomu obstarány, jak mohou být dále využity a jaký je celkový přínos pro daného uživatele aplikace, ať už se jedná o trenéra, nebo atleta. Vždy bude uvedeno, jaký typ dolovací metody byl potřeba a jaká se použila konkrétní metoda či algoritmus dolování. Mimo jiné bude také uvedeno, jakým způsobem bylo potřeba upravit (předzpracovat) vstupní data a jaké výsledky metody poskytují. Stručně bude popsáno, i proč byla pro řešení daného problému použita zrovna ta či ona metoda.

8.6.1 Značení segmentů

Během importu aktivit do systému je aktivita rozdělena na jednotlivé segmenty, jejichž obvyklá délka činí jeden kilometr. Poslední segment každé aktivity může mít délku odlišnou z toho důvodu, že celková délka aktivity nemusí být zaokrouhlena na celé kilometry. Takto extrahované segmenty každé importované aktivity jsou následně označeny, jinými slovy klasifikovány, dle jejich profilu, potažmo vývoje nadmořské výšky.

Z každého segmentu je tedy získán celkový výstup (v metrech) a celkový sestup (opět v metrech). Tato dvojice hodnot může být chápana jako bod, jehož souřadnice jsou právě výstup a sestup. Body jsou realizovány jako dvousložková pole, která jsou uložena do výsledného pole vstupní množiny algoritmu, který provádí značení. Každý bod je uložen na indexu daném hodnotou jednoznačného identifikátoru segmentu. Tímto způsobem jsou data připravena pro klasifikaci.

Dříve zmíněná knihovna PHP-ML nabízí pro klasifikaci tři dostupné algoritmy. Jsou jimi SVC (Support Vector Classification), algoritmus K nejbližších sousedů (K Nearest Neighbors) a Naive Bayes. Pro tento případ byl použit poslední zmiňovaný algoritmus. V průběhu testování a zkoumání byly vyzkoušeny všechny tři klasifikačni algoritmy. Byla vytvořena trénovací množina čítající zhruba 300 záznamů a také testovací množina o 50 záznamech. Každý z těchto algoritmů nejprve provedl trénovací část nad trénovacími daty a poté provedl ohodnocení testovací množniny. Testovací množnina byla také ohodnocena manuálně po konzultaci s dvěma atlety a vzniklo tedy referenční značení segmentů. Všechny tři agloritmy bez výjimky provedly značení bezchybně i při opakovaném testování. V tuto

	Úspěšnost značení [%]	Čas [s]
SVC	100	0.02237105
Naive Bayes	100	0.02401185
K Nearest Neighbors	100	0.00145888

Tabulka 8.2: Tabulka výsledků testování jednotlivých shlukovacích algoritmů

chvíli bylo učiněno rozhodnutí zkoumat algoritmy také z časového hlediska, kde si nejlépe vedl právě Naive Bayes. Výsledky testování je možné vidět v následující tabulce. Čas je v tomto případě brán jako průměr z deseti meření pro každý algoritmus.

Použitý algoritmus pro klasifikaci vyžaduje mimo vstupní sady dat, jejíž prvky bude klasifikovat (tvorba této sady je popsána v předchozím odstavci), také sadu trénovacích dat. Ta byla v tomto případě vytvořena a uložena do databázové tabulky *activity_segments_classification_train*. Každý záznam obsahuje celkový výstup, celkový sestup a označení profilu takového segmentu. Možnosti jsou:

- prudké klesání,
- klesání,
- rovina,
- zvlněný profil,
- stoupání a
- prudké stoupání.

Každý záznam v tabulce byl manuálně označen na základě subjektivního rozhodnutí tvůrce této práce, několika dotázaných atletů a také trenérů. Tato množina dat je před každou klasifikací formátována a předložena klasifikačnímu algoritmu a je provedena trénovací část. Následně je provedena samotná klasifikace, kde algoritmus Naive Bayes pro každý nově importovaný segment určí jeho značku, tedy profil.

Atlet či trenér zkoumající a analyzující absolvovanou aktivitu nemusí pro každý segment dohledávat celkový výstup a sestup a určovat, zda se jednalo o kilometr do kopce, po rovině, nebo z kopce, ale může využít tyto značky segmentů jako vodítka. Mimo tento přínos je klasifikace/značení segmentů aktivit využito také v další části dolování dat. Příklad přehledu již označených segmentů je možné vidět na obrázku **8.6**.

8.6.2 Podobně náročné trasy

Dalším úkonem při importu aktivit do systému, který se týká dolování dat a získaní znalostí z databází, je shlukování aktivit z hlediska náročnosti. Jedná se o poměrně zajímavou tréninkovou funkci systému. Každý uživatel má možnost u aktivity vidět seznam podobně náročných tras. Sportovec tedy může porovnat takové trasy z různých úhlů pohledu. Příkladem může být situace, kdy atlet absolvuje aktivitu v horách o určité délce a o určitém převýšení, aktivitu zaznamená do hodinek nebo chytrého telefonu a posléze ji nahraje do systému. Poté se může podívat na detail aktivity a kromě různých zajímavých metrik má možnost vidět podobně náročné absolvované trasy, pokud tedy nějaké takové již do systému importoval. Trasy může následně porovnat a zjistit, zda se zlepšil, zhoršil, kdy měl nejlepší

čas, kdy běžel v nejnižších tepech, kdy naopak v nejvyšších, kdy bylo jeho relativní úsilí nebo výkon (viz předchozí sekce 8.5) nejvyšší a tak dále.

Takové spojení nebo, dle úhlu pohledu, rozdělení tras aktivit do skupin je možné realizovat pomocí dolování dat, konkrétně pomocí shlukování (clustering). To bylo detailněji popsáno v kapitole 4. Velmi zjednodušeně a stručně řečeno se jedná a spojení objektů do shluků (cluster) dle jejich podobnosti.

Pro shlukování tras podle podobnosti byla z každé aktivity extrahována dvojice hodnot, vzdálenost a celkový výstup. Vzdálenost je v každé aktivitě uložena v kilometrech, například 15,4 km, přičemž celkový výstup v metrech. Aby při výsledném shlukování nehrál celkový výstup tak markantní roli, byla vzdálenost v rámci předzpracování vynásobena hodnotou 100. Přece jen jsou si více podobné dvě 20km trasy, kde na jedné výstup činí 200 metrů a na druhé 100 metrů, než jedna 10km trasa a jedna 20km, které mají totožný celkový výstup. Již zmíněná dvojice hodnot je následně umístěna do dvousložkového pole, které je vloženo do celkového pole množiny pro dolování dat na index odpovídající identifikátoru aktivity. Tím je dokončeno předzpracování. Výsledné body reprezentující aktivity je možné vidět na obrázku **8.7**. Obrázek je v této kapitole pouze v malém měřítku pro rychlý náhled a vytvoření představy čtenářem. Přehledněji ve velkém měřítku se obrázek nachází v příloze C. Jako reprezentativní vzorek bylo vzato 270 reálných aktivit systému.

Knihova pro strojové učení a dolování dat PHP-ML, která byla použita v této práci a byla také blíže popsána v sekci Použité technologie (8.1), poskytuje dva algoritmy pro shlukování. Jedná se o K-means a DBSCAN (popsány v kapitole 4). Oba tyto algoritmy byly testovány nad výše zmíněnou množinou vstupních dat. V případě algoritmu K-means byl nastaven parametr, který určuje počet výsledných shluků na hodnotu odpovídající počtu aktivit v množině krát 0,1. Hodnota parametru byla různě měněna, avšak nejlepší výsledky dával algoritmus právě s takto nastaveným parametrem. U algoritmu DBSCAN se zkoumáním a testováním dospělo k závěru, že nejlepší je v tomto případě zvolit velikost ε-okolí rovnu 6, a minimální počet aktivit v jednom shluku roven 2. Shlukování proběhlo nad stejnou množinou dat s využitím obou algoritmů a výsledky je možné vidět v grafech **8.8** a **8.9**. Opět se jedná pouze o malé obrázky a jejich varianty ve větším rozlišení se nachází v příloze C. Jednotlivé shluky jsou obarveny barvami pro názornost. Některé shluky, které se nachází v dostatečné vzdálenosti od sebe, aby nebyla možná jejich záměna, mají stejné barvy.

V případě algoritmu DBSCAN je možné vidět, že nejsou v grafu vykresleny aktivity, které nejsou součástí žádného shluku. V případě K-Means jsou vykresleny všechny aktivity. Po zkoumání a testování je možné říci, že DBSCAN působí o něco přísněji, a při této reprezentativní konfiguraci, která se však jeví jako nejlepší, zůstává mnoho aktivit mimo shluky. Oproti tomu K-Means je o něco více benevolentní a umožňuje větší odlišnosti v rámci jednoho shluku a tím také spojí více aktivit. Algoritmus K-Means má také pro toto použití přívětivější použití, vzhledem k tomu, že není nutné nastavovat jak minimální velikost shluků, tak hodnotu ε-okolí, kterou může být velmi těžké zvolit univerzálně tak, aby odpovídala rozdílným vstupním množinám. Pro hledání podobně náročných tras byl tedy použit algoritmus K-Means.

8.6.3 Trasy poblíž

Další částí aplikace, kde se využívá technik dolování dat, je hledání tras poblíž. Tento úkon je opět prováděn v okamžiku importu nových aktivit do systému. Atlet absolvuje nějakou aktivitu neboli trasu, nahraje ji do systému a pak má možnost vidět, zda nějaký sportovec,

který je součástí stejného týmu absolvoval trasu v blízkém okolí nově nahrané trasy. Atlet vidí nejen trasy poblíž, patřící jiným atletům, ale také svoje. Má možnost tedy zjistit, jestli běhá často na nějakém místě, popřípadě kolikrát již běžel v blízkosti určité oblasti. Se členy týmů, u kterých touto cestou zjistí, že absolvují aktivity poblíž, se může například domluvit na společném tréninku.

V tomto případě probíhá předzpracování dat následujícím způsobem. Každou aktivitu v tomto případě reprezentuje hned několik dvojic. Dvojici tvoří souřadnice zeměpisné šířky a délky. Díky záznamu GPS pro každou aktivitu a uložení vývoje zeměpisné polohy v čase v rámci záznamu aktivity v databázi je možné tyto informace extrahovat. Vzhledem k časové optimalizaci algoritmu není uvažován každý vzorek. Aktivita, jejíž trasa měří například 10 kilometrů, může mít takovýchto vzorků klidně 5000 nebo i více. V rámci měření a testování vyšlo najevo, že pro zachování kvality výsledků dolování dat není potřeba uvažovat každý vzorek. Počet vzorků, které se dostanou do výsledné vstupní množiny algoritmu, přímo závisí na délce aktivity. Pro aktivity menší něž 3 km se bere v potaz každý stý vzorek, pro aktivity v rozmezí 3 až 7 km každý dvoustý vzorek, pro aktivity v rozmezí 7 až 20 km každý pětistý a pro ostatní aktivity každý tisící. V rámci výsledné vstupní množiny algoritmu se každý vzorek nachází na indexu daném jednoznačným identifikátorem aktivity ve spojení s indexem daného vzorku v rámci aktivity.

Trasy poblíž dané trasy jsou hledány prostřednictvím shlukování. Jak již bylo napsáno dříve, použitá knihovna PHP-ML poskytuje dva algoritmy pro shlukování, kterými jsou K-Means a DBSCAN. V přechozí části kapitoly věnující se shlukování podobně náročných tras bylo znázorněno, jak se od sebe oba algoritmy liší. Byly uvedeny názorné grafy a bylo odůvodněno, proč byl použit právě algoritmus K-Means. Zde se situace trochu liší. Není nutné za každou cenu dostat trasu do nějakého shluku tak, že by ve výsledku byly například trasy označeny jako blízké, ačkoliv je dělí vzdálenost klidně 50 kilometrů. Z tohoto důvodu a také proto, aby bylo použití shlukovácích algoritmů v této práci více různorodé, byl pro řešení problému použit algoritmus DBSCAN. Výsledné shluky (clusters) jsou následně zpracovány a jsou z nich extrahovány reprezentující aktivity, které jsou si navzájem blízké. Vše je stejně jako v případě hledání stejně náročných tras uloženo do databáze.

8.6.4 Stanovení tréninkového profilu atleta

Jednou ze stěžejních vlastností aktuální verze aplikace je možnost stanovení tréninkového profilu atleta pro dané období. Tato funkčnost je dostupná pro jakýkoliv týden či měsíc, ve kterém má sportovec importovánu alespoň jednu aktivitu. Následně se mu na hlavním přehledu zobrazuje, jaké má zastoupení tréninkového profilu pro jednotlivé typy tras v daném období. Stejně tak této vlastnosti může využít trenér u svých sveřenců na stránce s jejich osobním přehledem, nebo také v týmovém přehledu, kde má možnost porovnání jednotlivých atletů. Procentuální zastoupení tréninkového profilu může být rozděleno mezi následující specializace:

- expert na stoupání,
- expert na prudká stoupání,
- expert na roviny,
- expert na zvlněný profil,
- expert na seběhy a

- expert na prudké seběhy.

Ukázka, jak takový přehled tréninkového profilu atleta pro jeden týden může vypadat, je na obrázku **8.10**.

Pro daného sportovce se z databáze získají všechny jeho aktivity v požadovaném období. Z nich se následně získají také jednotlivé segmenty o délce 1 km. Každý takový segment je již označen značkou určující jeho profil (stoupání, klesání, rovina, ...), jak již bylo uvedeno dříve. V segmentech jsou v tuto chvíli důležité dvě hodnoty. Jsou jimi relativní výkon (8.5.2) a relativní úsilí (8.5.3). Každý takový segment je reprezentován právě touto dvojicí hodnot, která je ve výsledné vstupní množině pro dolování dat umístěna na indexu odpovídajícímu identifikátoru aktivity a indexu segmentu v rámci aktivity. Mimo to se v rámci všech segmentů napříč aktivitami atleta v určeném období určí segment s nejvyšším relativním výkonem a zároveň nejnižším usilím. V podstatě je hledán segment, ve kterém měl sportovec nejvyšší relativní výkon, od kterého je odečteno relativní úsilí segmentu. Identifikátor tohoto segmentu je uložen v paměti v průběhu dolování dat.

Následně je provedeno shlukování segmentů reprezentovaných relativním výkonem a úsilím opět prostřednictvím algoritmu K-means. Pro výsledný tréninkový profil je žádnoucí, aby se ideálně téměř každý segment nacházel v nějakém shluku, a to i za cenu nižší kompaktnosti výsledných shluků. Také z tohoho důvodu byl vybrán právě algoritmus K-means. Po dokončení shlukování jsou procházeny jednotlivé výsledné shluky a je hledán ten, kde se nachází segment s nejvyšším výkonem za cenu nejnižšího úsilí. Poté jsou extrahovány všechny segmenty, které jsou také součástí tohoto shluku. Jedná se tedy o segmenty, kde daný atlet v daném období podával nejvyšší výkon při nejnižším vynaloženém úsilí. V těchto výsledných segmentech je spočítáno procentuální zastoupení jednotlivých profilů segmentů, a tím je zjištěno, na jakých profilech tratí se sportovci nejvíce dařilo. Z tohoto faktu může jak sportovec, tak jeho trenér získat velmi cenné informace, ke kterým může přihlédnou při plánování dalších tréninků, nebo účastí na závodech.

```php
/**
 * Download activity from Garmin by activity ID and store it as `.fit` file.
 * Returns `.fit` file path.
 *
 * @return string
 *
 * @throws GarminGetActivityAsFitFileFailedException
 */
public function handle(): string
{
    if (false === $this->validate()) {
        throw new GarminGetActivityAsFitFileFailedException('Validation failed');
    }

    try {
        // Auth
        $objGarminConnect = $this->authObject->authorize();
        // Download activity as .zip data
        $zipData = $objGarminConnect->getDataFile(
            GarminConnect::DATA_TYPE_FIT,
            $this->activityId
        );

        $zipFilePath = $this->getTmpStoragePath() . '/' . $this->activityId . '.zip';
        $this->fitFilePath = $this->getTmpStoragePath() . '/' . $this->activityId . '_ACTIVITY.fit';
        // Create .zip file from data
        file_put_contents($zipFilePath, $zipData);
        $zipFile = new ZipArchive;
        // Open .zip file and extract
        if (true === $zipFile->open($zipFilePath)) {
            $zipFile->extractTo($this->getTmpStoragePath());
            $zipFile->close();
        } else {
            throw new GarminGetActivityAsFitFileFailedException('Cannot open tmp .zip file.');
        }
        // Delete .zip file
        unlink($zipFilePath);
    } catch (Exception $e) {
        throw new GarminGetActivityAsFitFileFailedException($e);
    }
    return $this->fitFilePath;
}
```

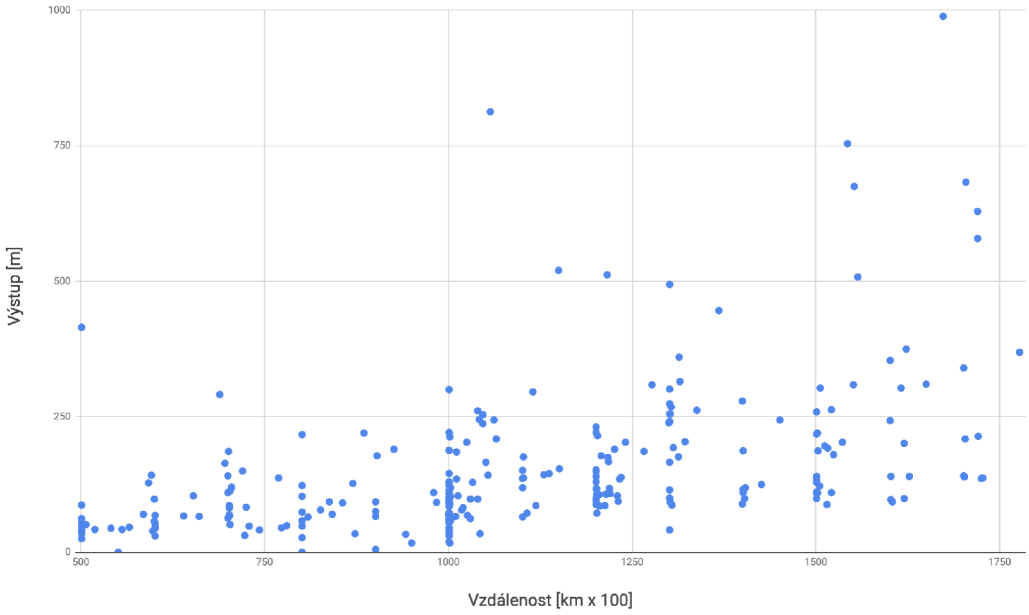
Obrázek 8.4: Příklad kódu implementace služby *GarminGetActivityAsFitFile*

```
➜  dip git:(master) ✗ php artisan my-training:garmin-import 1
[2021-03-21 20:42:37] Importing new activities.
➜  dip git:(master) ✗
```

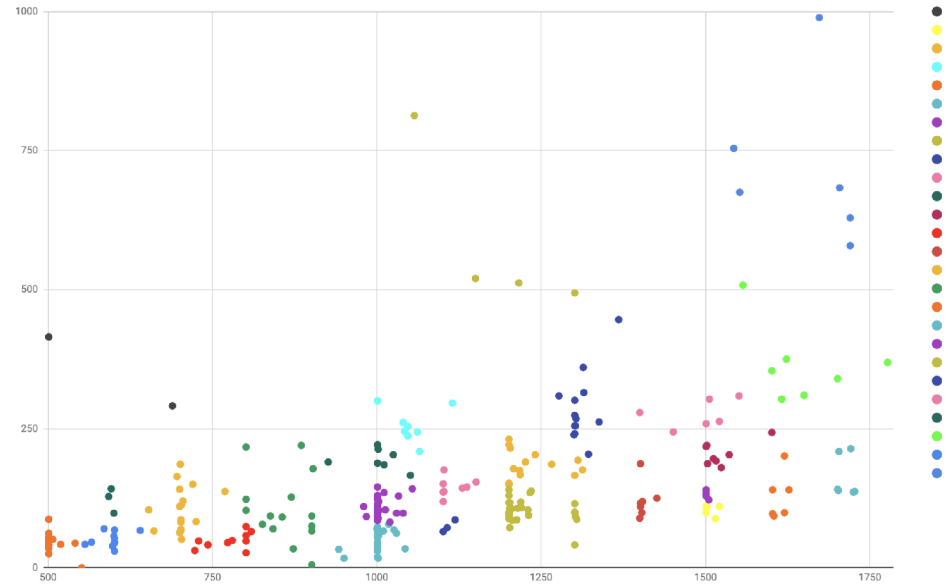
Obrázek 8.5: Příklad importu aktivit pomocí konzolového příkazu (atlet s id 1)

Vzdálenost	Profil	Tempo	Max. tempo	Tep	Max. tep	Výstup	Sestup	Max. výška
1 km	Stoupání	05:32 min/km	05:04 min/km	126 tzm	143 tzm	22 m	3 m	500 m
2 km	Rovina	05:05 min/km	04:41 min/km	141 tzm	146 tzm	4 m	21 m	503 m
3 km	Prudké klesání	04:51 min/km	04:39 min/km	147 tzm	160 tzm	0 m	85 m	482 m
4 km	Rovina	05:35 min/km	05:06 min/km	162 tzm	169 tzm	21 m	12 m	408 m
5 km	Stoupání	06:12 min/km	05:51 min/km	153 tzm	162 tzm	35 m	1 m	441 m
6 km	Stoupání	06:27 min/km	05:43 min/km	147 tzm	156 tzm	49 m	5 m	486 m
7 km	Klesání	05:52 min/km	05:31 min/km	150 tzm	162 tzm	0 m	45 m	485 m
8 km	Zvlněný profil	06:06 min/km	05:41 min/km	161 tzm	171 tzm	31 m	28 m	443 m

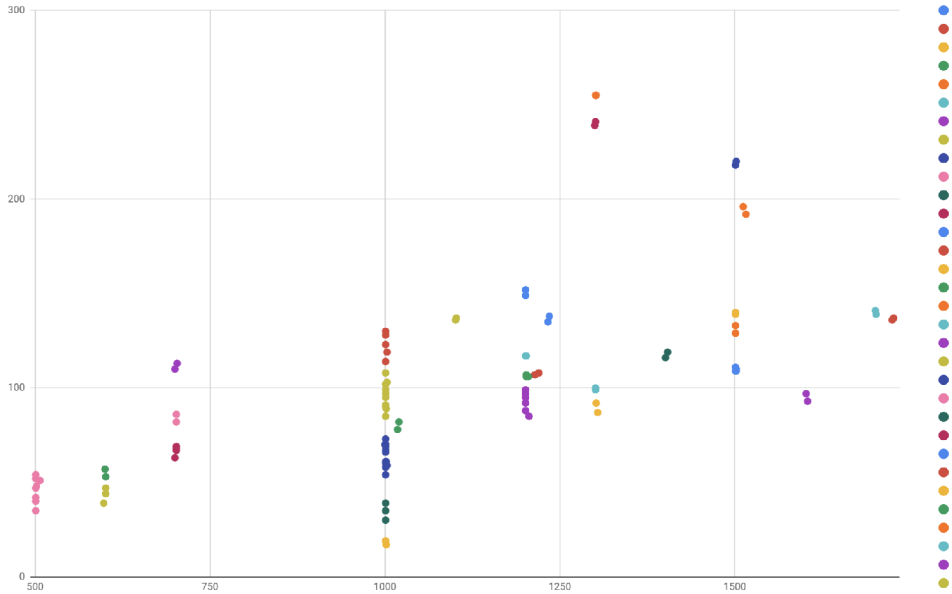
Obrázek 8.6: Příklad označených/klasifikovaných segmentů aktivity



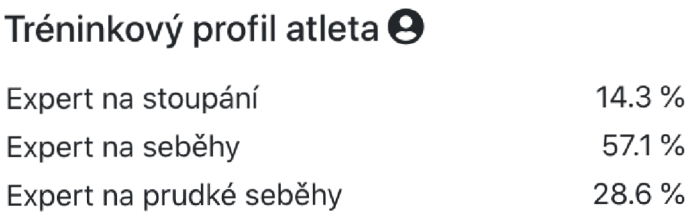
Obrázek 8.7: Graf bodové reprezentace aktivit (vzdálenost a celkový výstup aktivity)



Obrázek 8.8: Výsledek shlukování pomocí algoritmu K-Means



Obrázek 8.9: Výsledek shlukování pomocí algoritmu DBSCAN



Obrázek 8.10: Ukázka tréninkového profilu atleta

Kapitola 9

Testování

Doposud již byla popsána úplná analýza a návrh aplikace pro komplexní on-line tréninkový deník. Dále byly zmíněny použité technologie a některé zajímavé postupy při implementaci včetně vysvětlení využití a implementace dolování dat pro výpočet tréninkového profilu atleta. Poslední, avšak nedílnou součástí vývoje aplikací či systémů, je jejich testování. Zde bylo rozděleno do několika logických částí, které tvoří uživatelské testování, testování API rozhraní pro AJAX volání a v neposlední řadě testování algoritmů pro dolování dat.

V rámci uživatelského testování bylo nutné zjistit, zda systém splňuje požadavky definované v kapitole Analýza (6). Z této kapitoly vzešlo kromě slovního popisu očekávaných výstupů implementace jednotlivých částí a aspektů aplikace také několik schémat obsahujících diagramy případů užití. Zároveň zde bylo využito praktické zkušenosti několika hobby sportovců, atletů a trenérů, kteří představili své požadavky na systém tohoto typu a zmínili, co jim ve stávajících dostupných řešeních chybí. Ti stejní lidé se tedy logicky podíleli i na uživatelském testování systému za účelem ověření, zda nová aplikace splňuje jejich požadavky, případně do jaké míry. Nutno dodat, že většina z těchto účastníků fáze analýzy a následně i testování sdělovala svoje připomínky a nápady na úpravy již během fáze implementace a bylo tím zaručeno, že se vývoj aplikace neodkloní dramatickým způsobem od požadovaných cílů. Jeden ze stěžejních odstavců (6.9) kapitoly Analýza je následující.

> Dalším problémem jsou tréninkové skupiny a týmy trénované trenéry. Trenéři potřebují podrobné statistiky o svých svěřencích. Zejména v dnešní komunikačně vyspělé době dochází k tomu, že trenéři a jejich svěřenci se příliš často nepotkávají, nebo se neviděli nikdy a trénovaní probíhá na dálku. Například spousta českých ultra-maratonců využívá služby trenérů z Rakouska, USA či jiných zemí. Ti potřebují podrobná data o tom, jak se jejich svěřenci aktuálním tréninkem profilují a které atlety vybrat a nominovat na nadcházející závody. Zde již aplikace jako Strava, Garmin Connect, nebo tabulky v programu Excel, nebo dokonce sdílené dokumenty nestačí. To a další aspekty a problémy by měla řešit aplikace vyvíjená v rámci této diplomové práce.

Zmíněný fakt byl vyřešen implementací funkce tréninkového profilu atleta, který je možné zobrazit jak individuálně, tak v týmovém kontextu. Všichni dotázaní se shodli na tom, že hlavní cíl celé aplikace byl splněn a v následujících měsících se ukáže, zda se o tato data dá reálně opřít při plánování příštích tréninků. Dále byli také všichni účastníci testování požádáni, zda by mohli sepsat své připomínky k aplikaci. Objevilo se jich pouze několik málo, zejména ke grafické stylizaci, načež po následné diskuzi byla velká většina z nich zapracována.

Pro zobrazení mapových podkladů a trajektorií aktivit bylo nutné vytvořit koncový bod API. Stejně tak tomu bylo v případě načítání detailů o segmentech aktivit v jejich modálních oknech na detailu konkrétní aktivity. Byly tedy vytvořeny dva koncové body, které bylo nutné také otestovat. První bod pro získání souřadnic trajektorie vyžaduje jako parametr jednoznačný identifikátor aktivity. Pokud aktivita není nalezena, je vrácen kód 404, pokud aktivita neobsahuje pole se souřadnicemi, pak bod vrací kód 400. Pokud vše proběhne v pořádku, je vráceno pole objektů obsahujících dvojice souřadnic a návratový kód 200. Podobný princip je uplatněn i ve druhém případě. Pro otestování těchto koncových bodů byl využit nástroj Postman[1], díky kterému bylo možné testovat chování při korektních i chybných vstupech. V průběhu testování byla objevena pouze drobná chyba, která byla ihned opravena.

Nedílnou součástí fáze testování bylo zkoumání a testování dolovacích algoritmů. Tato fáze se přímo prolínala s fází implementace a je možné říci, že probíhaly takřka paralelně. Protože je tato problematika blíže probrána v sekci Dolování dat (8.6) v kapitole Implementace a také přímo v samostatné kapitole Dolování dat (4), není nutné se zde příliš rozepisovat. Byly analyzovány a testovány jednotlivé algoritmy dostupné v knihovně PHP-ML. V případě klasifikačních algoritmů byla testována jejich spolehlivost, to znamená, s jakou přesností poskytují výsledky při použití specifickém pro tuto aplikaci. Dále byly také algoritmy testovány z hlediska rychlosti. Při testování spolehlivosti se dospělo k závěru, že všechny dostupné algoritmy jsou naprosto dostačující a rozhodujícím faktorem se tedy stala rychlost vyhodnocení. V případě shlukovacích algoritmů byla naopak testována kompaktnost jednotlivých shluků a spíše než na správnost či chybnost algoritmů, byl kladen důraz na vhodnost pro řešení jednotlivých problémů. Jeden algoritmus se hodil na shlukování podobně náročných tras, přičemž jiný na shlukování tras v blízkosti.

V práci bylo uvažováno také nad implementací jednotkových (unit) testů pro datovou vrstvu aplikace. V tomto případě byl ale použit Laravel ORM Eloquent, který ošetřuje všechny možné chybové stavy a situace a je kompletně testován vždy před vydáním nové verze, a to přímo vývojáři Laravel. Vzhledem k tomu, že veškeré úkony na datové vrstě jsou realizovány pomocí této architektury, nebylo nutné vytvářet takto orientované testy.

[1]Postman - https://www.postman.com

Kapitola 10

Závěr

Cílem této diplomové práce bylo navrhnout a implementovat komplexní on-line tréninkový deník pro sportovce. Nejprve bylo nutné si nastudovat, nebo alespoň osvěžit některé základní pojmy týkající se tvorby aplikací a informačních systémů. V tomto případě se jedná o webovou aplikaci, případně webový informační systém, a tak bylo nutné si osvojit principy a technologie týkající se tohoto odvětví včetně databází, nebo frontend a backend částí. Vzhledem k zaměření aplikace bylo nasnadě uvést a popsat určité sportovní pojmy související s danou problematikou. On-line tréninkový deník potřebuje v první řadě odněkud získávat data o sportovních aktivitách uživatelů. Tomu byla věnována celá druhá kapitola této práce. Bylo analyzováno, jaké technologie současní sportovci používají a jaké jsou možnosti importu dat. K tomu byl využit také dotazník dostupný v přílohách. Jednou z klíčových částí bylo použití dolování dat a s tím souvisejících algoritmů pro analýzu sportovních dat a vyvození učitých netriviálních závěrů. Pro tyto účely byla uvedena a podorobněji rozepsána značná část problematiky dolování dat, včetně shlukování a klasifikace. Poté byla dohledána a porovnána již existující řešení a byly určeny jejich slabiny či silná místa, ze kterých je možné těžit při tvorbě nové aplikace. Dalším krokem bylo provedení analýzy toho, co uživatelé aplikací tohoto typu potřebují a čeho se jim ve stávajících řešeních nedostává.

Z této fáze vzešly určité cíle definující podstatu výsledného systému. Výstupem kapitoly Analýza byly jednotlivé diagramy případů užití. Ještě před implementací bylo potřeba klíčové části systému důsledně navrhnout. Hlavní důraz byl kladen na návrh databáze a datových tříd. Výstupem bylo několik ER diagramů pro různé iterace vývoje a také návrh datových tříd a v neposlední řadě návrh onoho importu sportovních dat atletů do systému. Vzápětí již následovala implementace, kde byla představena a podrobněji rozebrána tvorba zásadních částí, jako třeba část pro manuální a automatický import, výpočet pokročilých sportovních metrik v rámci předzpracování dat a zejména samotné dolování dat, do kterého spadá značení segmentů aktivit, shlukování podobně náročných tras, tras poblíž a stanovení tréninkového profilu atleta. Hlavně poslední zmíněná součást představuje jednu z hlavních náplní celé práce, díky které byl splněn jeden ze stanovených cílů. Přitom spojuje předchozí uvedené časti v jednu. Představeny byly také použité technologie. Namátkou se jednalo o Laravel, MySQL, PHP-ML a další. Po fázi implementace byl výsledek otestován ať už uživatelským testováním, nebo například testováním vhodnosti použití jednotlivých dolovacích algoritmů pro danou problematiku.

Výsledek diplomové práce má tedy poměrně široké spektrum využití. Je vhodný pro potenciální začínající sportovce, kteří se rozhodli nějakým způsobem zaznamenávat své sportovní aktivity a jsou jednoduše zvědaví, zda se zlepšují, jak se jim dařilo v předchozím

období, nebo zda podávají lepší výkony do kopce, nebo třeba po rovině. Stejně tak najde aplikace zcela jistě využití v rukou pokročilejších sportovců, kterým aktuální dostupné aplikace nevyhovují například z důvodu, že nemají k dispozici týdenní, měsíční nebo roční souhrny, nebo je zajímá, jaký je jejich tréninkový profil za určitá obodobí a jakým směrem se mají dále v tréninku ubírat. Hlavní skupinu potenciálních uživatelů však tvoří pokročilejší sportovci či atleti, kteří se chtějí dále zlepšovat a to i s použitím moderních technologií, jakými jsou například dolování dat. Ve spolupráci s trenéry a pomocí této aplikace mají možnost dosáhnout efektivně svých cílů a podávat kvalitní výkony díky tréninku založeném třeba na vývoji tréninkového profilu stanoveného pomocí této aplikace. Stejně tak tomu je v případě týmů.

Do budoucna by bylo možné dále zapracovat na možnostech importu sportovních aktivit do systému. Jednou z možností je například rozšíření počtu podporovaných formátů souborů, nebo napojení na aplikační rozhraní dalších výrobců záznamových zařízení, či napojení na aplikace třetích stran, například na aplikaci Strava. Aktuálně je zvažována také možnost propojení systému se systémy pro měření HRV (Heart Rate Variability - Variability srdečního tepu) a možnost přímo zobrazovat jak efekt jednotlivých tréninků, tak také výsledky regeneračních procesů a další zajímavá data. V neposlední řadě by se další vývoj mohl zaměřit na tvorbu mobilní verze aplikace.

Literatura

[1] Berka, P. *Dobývání znalostí z databází.* Academia, 2003.

[2] Chaudhary, M. a Kumar, A. *PhpStorm Cookbook.* Packt Publishing Ltd, 2014.

[3] Crockford, D. *JavaScript: The Good Parts: The Good Parts.* "O'Reilly Media, Inc.", 2008.

[4] Da Rocha, H. *Learn Chart. js: Create interactive visualizations for the Web with Chart. js 2.* Packt Publishing Ltd, 2019.

[5] DuBois, P. *MySQL.* Pearson Education, 2008.

[6] Han, J., Kamber, M. a Pei, J. *Data mining : concepts and techniques. Second Edition.* Elsevier Inc., 2006. ISBN 1-55860-901-3.

[7] Hruška, T. a Křivka, Z. *Informační systémy Studijní opora – Fakulta informačních technologií VUT v Brně.* [Online; navštíveno 29.12.2020]. Dostupné z: https://www.fit.vutbr.cz/study/courses/WAP/private/podklady/Opory/OIISPojem.pdf.

[8] Jacksi, K. a Abass, S. Development History Of The World Wide Web. *Int. J. Sci. Technol. Res.* 2019, sv. 8, s. 75–79.

[9] Koop, J. a Rutberg, J. *Training Essentials for Ultrarunning: How to Train Smarter, Race Faster, and Maximize Your Ultramarathon Performance.* VeloPress, 2016. ISBN 9781937716806.

[10] Lundby, C., Montero, D. a Joyner, M. Biology of VO2max: looking under the physiology lamp. *Acta Physiologica.* Wiley Online Library. 2017, sv. 220, č. 2, s. 218–228.

[11] Nedelcu, C. *Nginx HTTP Server: Adopt Nginx for Your Web Applications to Make the Most of Your Infrastructure and Serve Pages Faster Than Ever.* Packt Publishing Ltd, 2010.

[12] Novotnỳ, J. a Novotná, M. Fyziologické principy tréninku a testy běžců. *Atletika, ročník 60, 2008, číslo 11, doplněk „atletika plus “, str. 1-5 a.* 2008, sv. 8, s. 1–5.

[13] The editors of Runner's World. *The Best Running Apps to Take on Your Workout.* [Online; navštíveno 02.01.2021]. Dostupné z: https://www.runnersworld.com/gear/a20865699/best-running-apps.

[14] Somasundaram, R. *Git: Version control for everyone.* Packt Publishing Ltd, 2013.

[15] Spurlock, J. *Bootstrap: Responsive Web Development*. "O'Reilly Media, Inc.", 2013.

[16] Stauffer, M. *Laravel: Up & Running: A Framework for Building Modern PHP Apps*. O'Reilly Media, 2019.

[17] Thakur, R. N. a Pandey, U. The Role of Model-View Controller in Object Oriented Software Development. *Nepal Journal of Multidisciplinary Research*. 2019, sv. 2, č. 2, s. 1–6.

[18] Welling, L. a Thomson, L. *PHP a MySQL: rozvoj webových aplikací*. SoftPress, 2003. ISBN 80-86497-60-7.

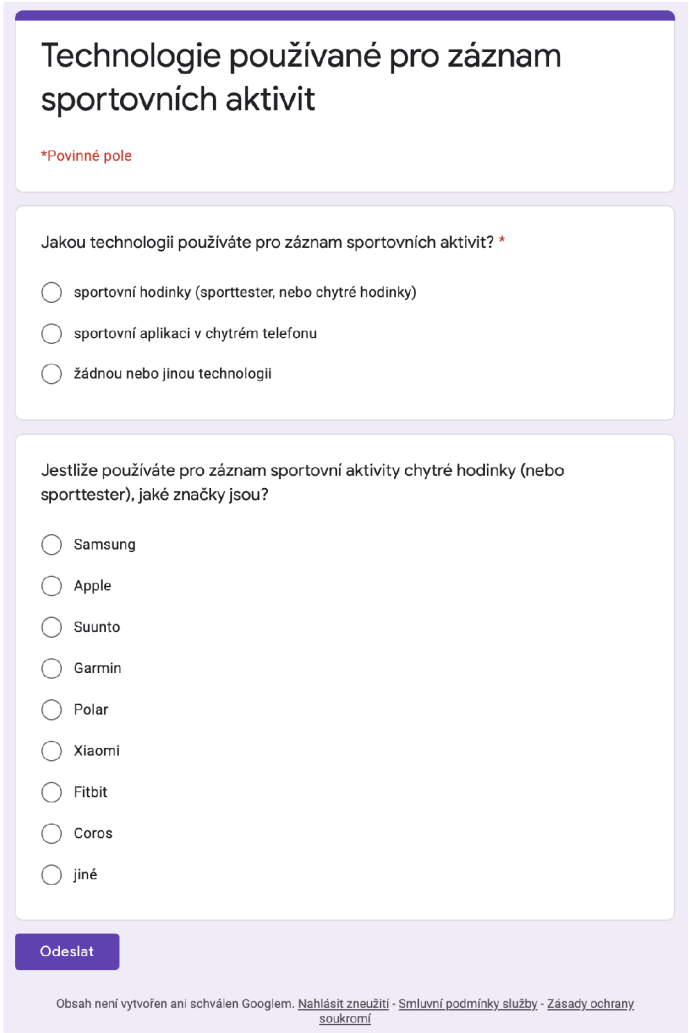
[19] White, M. *Is PHP Now Suitable For Machine Learning?* November 2015. [Online; posted 25-November-2015]. Dostupné z: https://medium.com/@syntheticmatt/is-php-now-suitable-for-machine-learning-a24e0f3233ac.

[20] Zemčík, P. *Tvorba uživatelských rozhraní Studijní opora – Fakulta informačních technologií VUT v Brně*. [Online; navštíveno 30.12.2020]. Dostupné z: https://wis.fit.vutbr.cz/FIT/st/course-files-st.php?file=%2Fcourse%2FITU-IT%2Ftexts%2FlTU-Podpora.pdf&cid=12203.

[21] Zendulka, J., Bartík, V., Lukáš, R. a Rudolfová, I. *Získávání znalostí z databází Studijní opora – Fakulta informačních technologií VUT v Brně*. [Online; navštíveno 03.01.2021]. Dostupné z: https://wis.fit.vutbr.cz/FIT/st/cfs.php?file=%2Fcourse%2FZZN-IT%2Ftexts%2FZZN.pdf&cid=13539.

Příloha A

Dotazník - technologie používané pro záznam sportovních aktivit

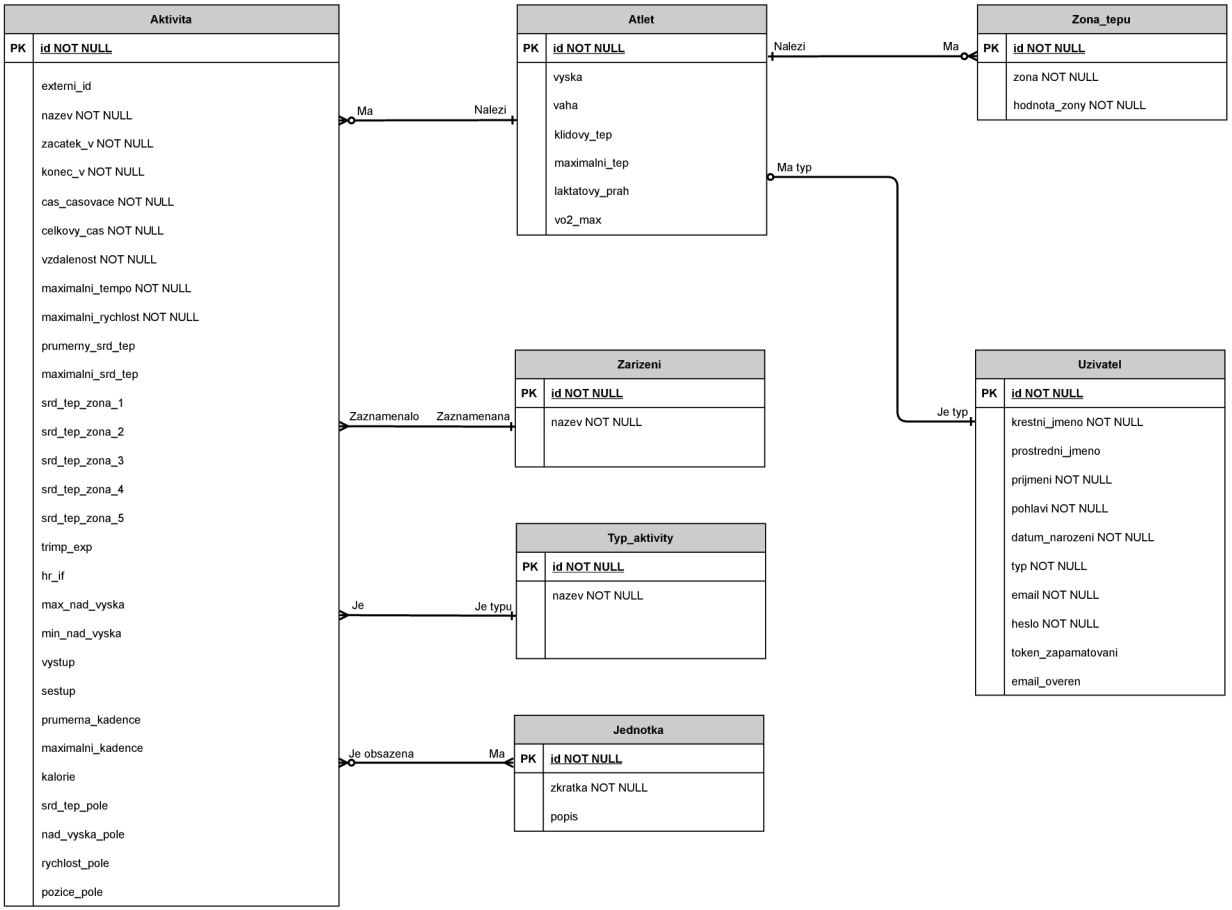


Obrázek A.1: Dotazník - technologie používané pro záznam sportovních aktivit

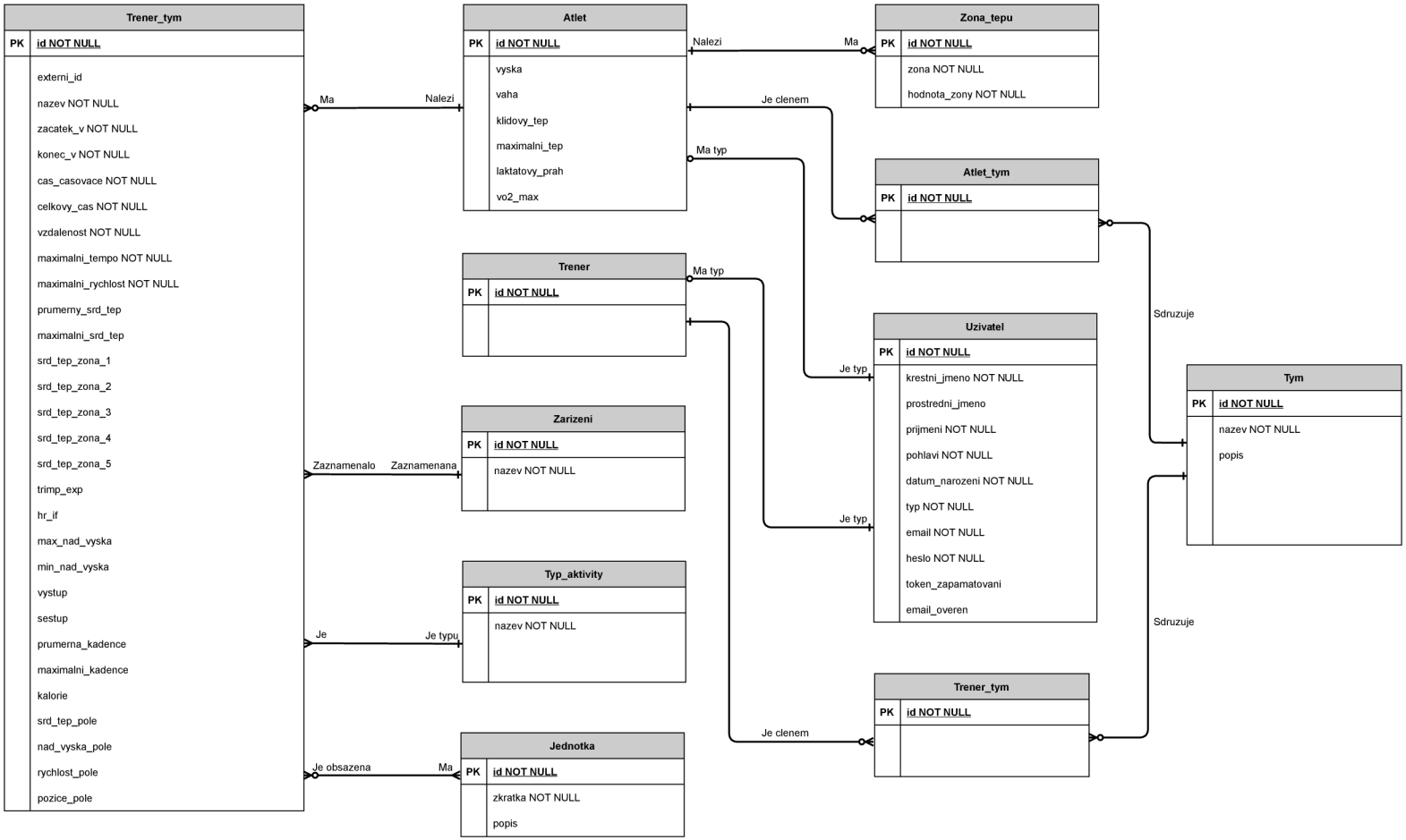
Příloha B

ER diagramy jednotlivých iterací

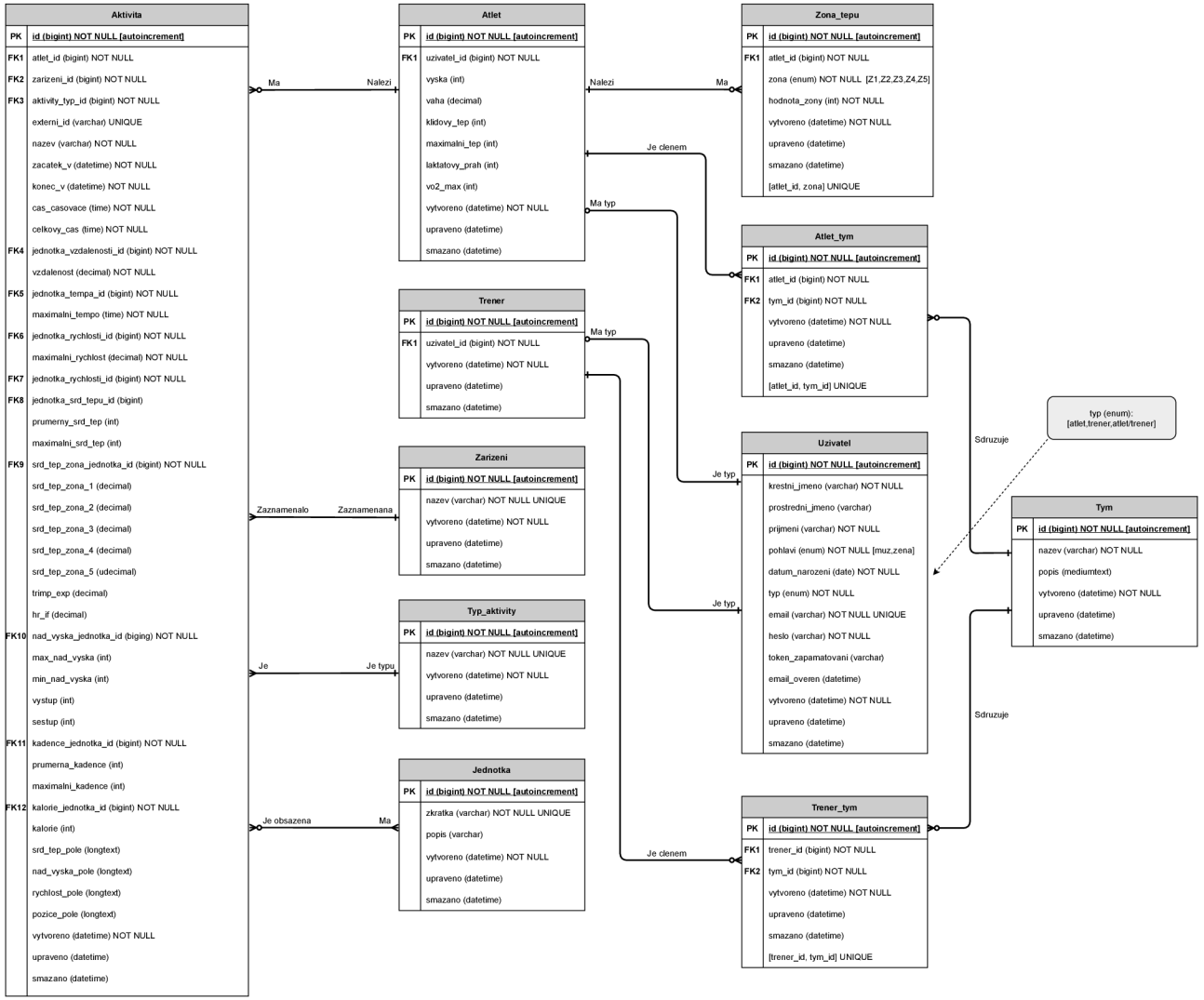
V této příloze jsou obsaženy ER (Entity Relationship) diagramy pro všechny tři iterace návrhu entit systému, ze kterých následně vychází samotná databáze systému, počínaje první iterací.



Obrázek B.1: ER diagram první iterace



Obrázek B.2: ER diagram druhé iterace

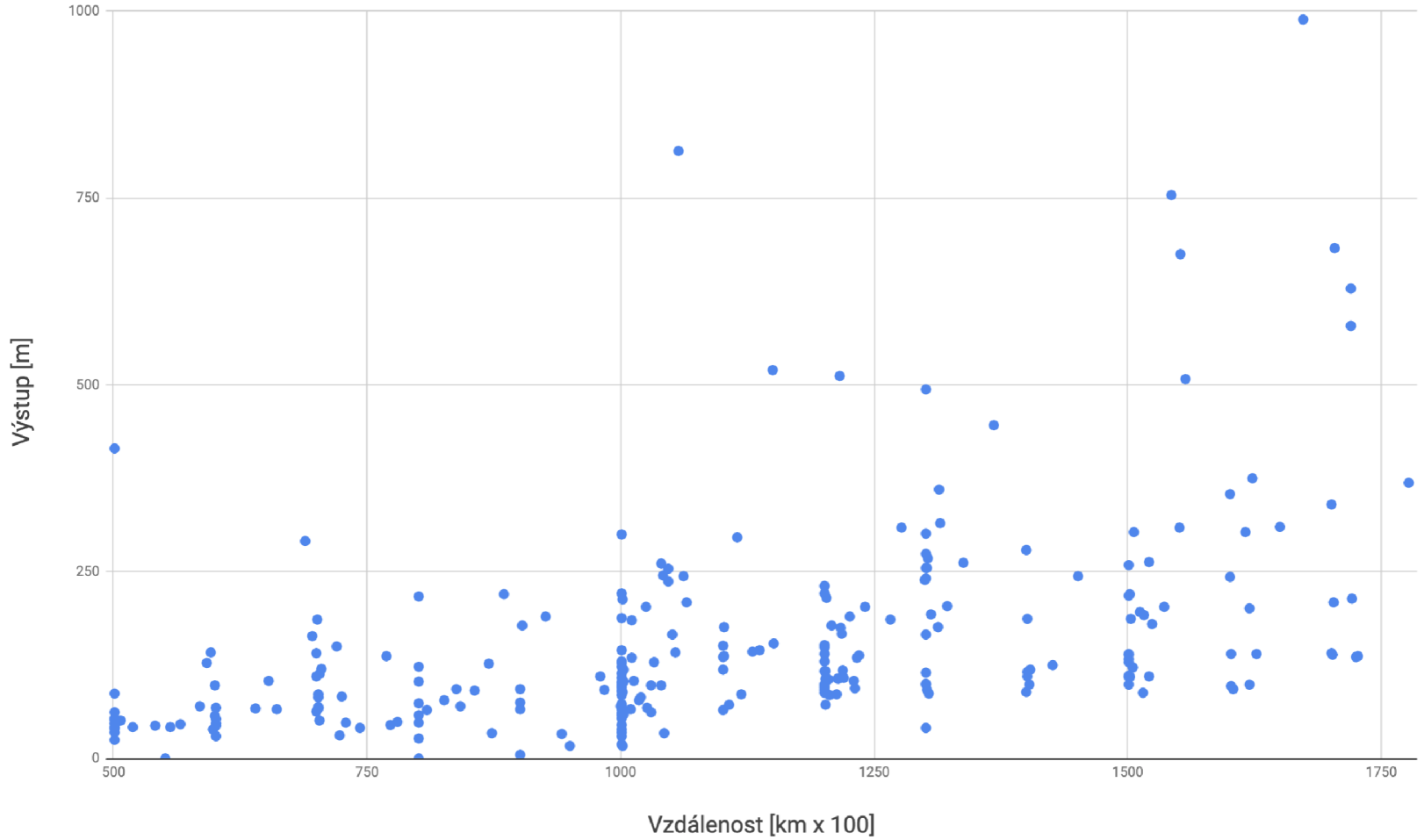


Obrázek B.3: ER diagram třetí (finální) iterace

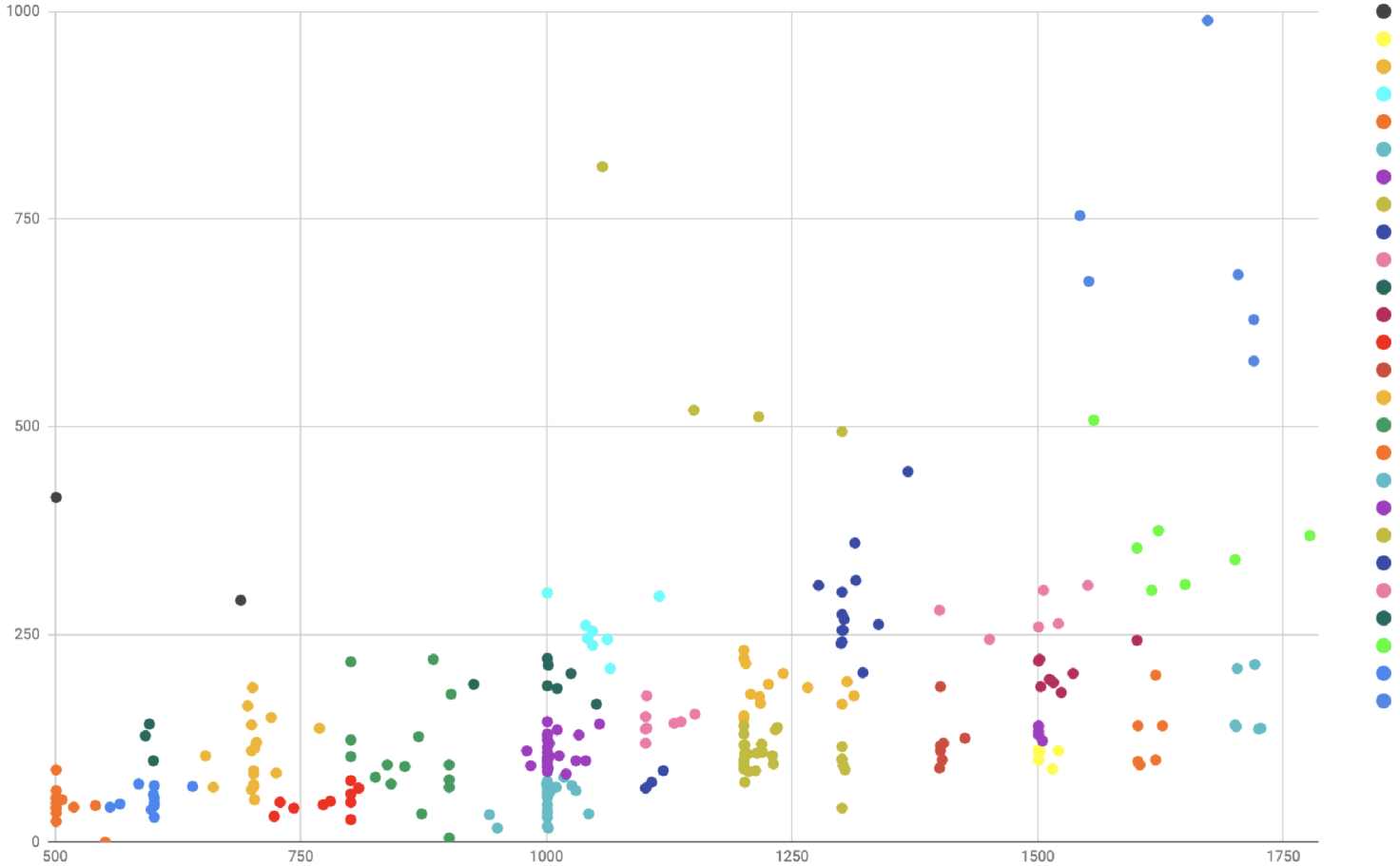
Příloha C

Shlukování

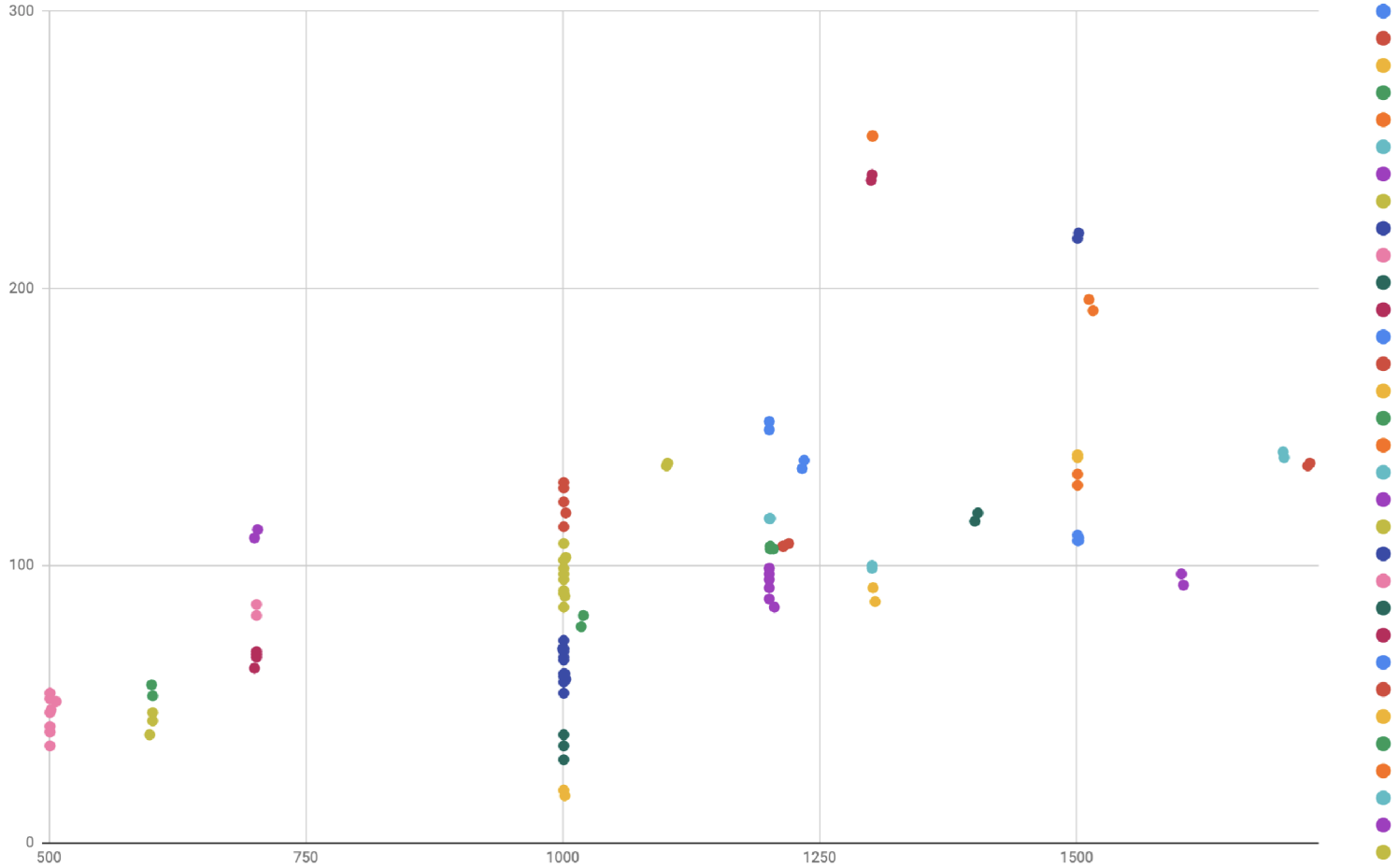
V této příloze se nejprve nachází graf vstupní množiny pro shlukování aktivit dle jejich náročnosti, kde každý bod odpovídá jedné aktivitě, přičemž souřadnice reprezentují vzdálenost aktivity (osa x) a celkový výstup aktivity (osa y). Dále se zde nachází grafy pro výsledky shlukování aktivit dle jejich podobnosti prostřednictvím algoritmů K-Means a DBSCAN. Jednotlivé shluky jsou odlišeny barvami.



Obrázek C.1: Graf bodové reprezentace aktivit (vzdálenost a celkový výstup aktivity)



Obrázek C.2: Výsledek shlukování pomocí algoritmu K-Means



Obrázek C.3: Výsledek shlukování pomocí algoritmu DBSCAN