Univerzita Hradec Králové
Fakulta informatiky a managementu
Katedra informatiky a kvantitativních metod

Aplikace pro sledování kalorického příjmu na platformě iOS

Diplomová práce

Autor: Daniel Krejčí
Studijní obor: Aplikovaná informatika

Vedoucí práce: doc. Ing. Filip Malý, Ph.D.

Hradec Králové Duben 2023

Prohlášení:

Prohlašuji, že jsem diplomovou práci zpracoval samostatně a s použitím uvedené literatury.

V Hradci Králové dne 21.4.2023 Daniel Krejčí

Poděkování:

Děkuji vedoucímu diplomové práce doc. Ing. Filipu Malému, Ph.D., za konzultace a odborné vedení při tvorbě práce. Děkuji také rodině a blízkým za podporu během studia.

Anotace

Diplomová práce se zabývá způsoby, jak mohou moderní technologie z oblasti mobilních aplikací, umělých neuronových sítí a počítačového vidění pomoci lidem sledovat jejich kalorický příjem a tím podpořit udržení zdravého životního stylu.

První kapitoly se věnují úvodu do problematiky sledování kalorického příjmu a popisují současná dostupná řešení na trhu. Dále se zaměřují na principy fungování umělých neuronových sítí a podrobně popisují softwarové technologie použité při implementaci navrženého řešení. Praktická část práce se zabývá návrhem aplikace, použitou metodiku vývoje a postupem implementace. Závěr je věnován průběhu testování, shrnutí dosažených výsledků a celkovému zhodnocení práce.

Výsledná mobilní aplikace na platformě iOS využívá vlastní model umělé neuronové sítě pro identifikaci a rozpoznání potravin s využitím frameworku CoreML a nástroje CreateML. Navržená databáze obsahuje informace o kalorických a výživových hodnotách potravin, umožňuje fulltextové vyhledávání podle názvu potraviny a následné zapsání do jídelníčku uživatele.

Annotation

This diploma thesis explores how modern technologies ranging from mobile applications to artificial neural networks and computer vision can help people monitor their calorie intake and thereby support the maintenance of a healthy lifestyle.

The first couple of chapters provide an introduction to calorie intake monitoring and describe the solutions currently available on the market. Furthermore, they outline the principles of artificial neural networks and describe in detail the software technologies used to implement the proposed solution. The practical part of the thesis covers the application design, the development methodology used, and the

implementation procedure. The final part details the testing process, summarizes the results, and provides an overall evaluation of the project.

The resulting mobile application built on the iOS platform uses a fully custom artificial neural network model built to identify and recognize food items using the CoreML framework and the CreateML toolkit. The designed backend database contains data about the caloric and nutritional values of various food types, enables full-text search based on the food name, and enables the user to enter it into their saved menu list.

Title: Calorie tracking application for iOS platform

Obsah

Seznam obrázků

Seznam tabulek

Seznam rovnic

1 Úvod

Podle dat Českého statistického úřadu trpí 18,5 % lidí v České republice obezitou a až 47 % mužů a 33 % žen trpí lehkou nadváhou [1]. Není proto překvapivé, že se mnoho z nás rozhodne začít v určitém okamžiku života hubnout. Motivací může být zlepšení zdraví, více energie, cítit se psychicky lépe, zvýšení sebevědomí, více si užívat běžných aktivit nebo být atraktivnější. Nicméně mnoho lidí se do hubnutí vrhne hlavou napřed aniž by si celý proces naplánovali a prostudovali a často se potom stane, že člověk začne hubnout, a nakonec skončí s ještě horší postavou, než s jakou začal. To může být frustrující zklamání, což některé lidi vede k tomu, aby se vzdali svých snah úplně. Je důležité k hubnutí přistupovat zdravě, udržitelným způsobem, s dobře naplánovaným procesem, nezbytnými znalostmi a podporou. Jedině tak lze zajistit dlouhodobý úspěch.

Zdravý životní styl je soubor návyků a postojů, které se týkají našeho zdraví a kvality života. Zahrnuje různé aspekty, jako je stravování, pohyb, odpočinek, psychická pohoda, kvalita vztahů a životní prostředí. Cílem zdravého životního stylu je zlepšit a udržet si dobré zdraví a pohodu a zabránit vzniku různých onemocnění. To lze dosáhnout například tím, že budeme jíst vyváženou a pestrou stravu, pravidelně se hýbat, dostatečně odpočívat, řešit stresové situace a vyhýbat se škodlivým návykům, jako je kouření nebo nadměrné pití alkoholu.

Kalorický příjem je jedním z hlavních faktorů, které ovlivňují váhu. Pokud přijmeme více kalorií, než kolik spálíme, pravděpodobně přibereme na váze. Naopak, pokud spálíme více kalorií, než kolik přijmeme, pravděpodobně zhubneme.

Zapisování kalorií umožňuje sledovat, kolik kalorií přijmeme, a pomáhá lépe plánovat stravu tak, abychom mohli dosáhnout svého cíle. Můžeme například snížit příjem kalorií tím, že nahradíme některé vysoce kalorické potraviny zdravějšími možnostmi nebo tím, že omezíme množství jídla, které jíme. Zapisování kalorií také může pomoci zlepšit stravovací návyky a vyhnout se nezdravému jídlu. Když si uvědomíme, kolik kalorií obsahuje určité jídlo, můžeme se rozhodnout, zda je pro nás vhodné ho jíst, nebo zda bychom měli raději zvolit zdravější variantu.

Diplomová práce se zabývá způsoby, jak mohou moderní technologie z oblasti mobilních aplikací, umělých neuronových sítí a počítačového vidění pomoci lidem sledovat jejich kalorický příjem a tím podpořit udržení zdravého životního stylu.

První kapitoly jsou věnovány úvodu do problematiky sledování kalorického příjmu a popisují současná dostupná řešení na trhu. Dále se zaměřují na principy fungování umělých neuronových sítí a podrobně popisují softwarové technologie použité při implementaci navrženého řešení. Praktická část práce se zabývá návrhem aplikace, použitou metodiku vývoje a postupem implementace. Závěr je věnován průběhu testování, shrnutí dosažených výsledků a celkovému zhodnocení práce.

2 Cíl práce

Cílem práce je vytvořit mobilní aplikaci na platformě iOS pro sledování kalorického příjmu a podpoření zdravého životního stylu s využitím dostupných moderních technologií z oblasti mobilních aplikací, umělých neuronových sítí a počítačového vidění.

Aplikace by měla využívat vlastní model umělé neuronové sítě pro identifikaci a rozpoznání potravin. Předpokládá se využití frameworku CoreML a nástroje CreateML, které jsou součástí platformy iOS.

Dílčím cílem je analýza stávajících aplikací na trhu. Zjištění všech dostupných funkcí umožní získat ucelený pohled o konkurenčním prostředí a identifikovat potencionální nedostatky a příležitosti pro inovace.

Dalším důležitým krokem je analýza a volba vhodných nástrojů a technologií, které by měly být moderní, rychlé a vhodné pro realizaci aplikace.

Nezbytnou součástí práce je návrh architektury, která aplikace zajistí plynulý chod a efektivní komunikaci mezi jednotlivými částmi.

Aplikace by měla disponovat přehledným uživatelských rozraním, které poskytne prostředí pro vyhledávání v databázi a zapisování potravin do jídelníčku. Novým uživatelům je vytvořen individuální plán pro příjem kalorií a makroživin. Nezbytnou funkcí jsou následně statistiky a doporučení, které uživateli poskytnout kontrolu nad svým zdravím a kondicí.

3 Pojmy v oblasti kalorií

Následující kapitola přibližuje a vysvětluje základní pojmy související s tématem zdravého životního stylu, které jsou nezbytné pro pochopení konceptu aplikace.

3.1 Kalorie

Kalorie je jednotka energie označovaná jako *cal*, která se používá k vyjádření energetické hodnoty potravin. V kontextu výživy se jedná o energii, kterou lidské tělo potřebuje k udržení všech životně důležitých funkcí a k provádění fyzických aktivit, během kterých tuto energii spotřebovává. Francouzský chemik Henri Vicrot Regnault definoval kalorii jako množství energie potřebné ke zvýšení 1 g vody z 0 °C na 1 °C [2].

3.2 Index tělesné hmotnosti

Index tělesné hmotnosti (BMI) je poměr mezi váhou a výškou těla (1) [3]. Je to jednoduchý způsob, jak zjistit, zda je tělesná váha v normálním rozmezí vzhledem k výšce. BMI se obvykle používá jako indikátor možného rizika zdravotních problémů spojených s nadváhou nebo podváhou (Tabulka 1).

$$BMI = \frac{hmotnost\ (kg)}{(výška\ (m))^2} \qquad (1)$$

Tabulka 1- Reprezentace výsledné hodnoty výpočtu BMI [3]

BMI	**Kategorie**
18,5	Podváha
18,5 až 24,9	Normální váha
25,0 až 29,9	Nadváha
30,0 až 34,9	Obezita třídy 1
35 až 39,9	Obezita třídy 2
40,0 a vyšší	Obezita třídy 3

3.3 Bazální metabolismus

Bazální metabolismus (BMR) je množství energie, které tělo vynakládá v klidovém stavu na základní životní funkce jako je dýchání, krevní oběh, trávení a udržování teploty těla. Bazální metabolismus tvoří 60 až 75 % celkového denního energetického výdeje člověka [4]. Výše bazálního metabolismu závisí na věku, pohlaví, tělesné hmotnosti, výšce a množství svalové hmoty. Lidé s vyšším podílem svalové hmoty obvykle mají vyšší bazální metabolismus, protože svaly spotřebovávají více energie než tuk. Je důležité, aby byl bazální metabolismus vyvážený s celkovým energetickým příjmem a výdejem, aby se zabránilo nadváze nebo podvýživě [5].

3.3.1 Harris-Benedictova rovnice

Harris-Benedictova rovnice je matematický vzorec, který se používá k výpočtu přibližné hodnoty BMR člověka pomocí hodnot pohlaví, hmotnosti, výšky a věku. Harris-Benedictova rovnice byla vyvinuta v roce 1919 a upravena v roce 1984 [6].

$$BMR\ ženy = 655 + (9{,}6 \times hmotnost\ kg) + (1{,}8 \times výška\ cm) - (4{,}7 \times věk\ v\ letech) \qquad (2)$$

$$BMR\ muži = 66 + (13{,}7 \times hmotnost\ kg) + (5 \times výška\ cm) - (6{,}8 \times věk\ v\ letech) \qquad (3)$$

Výsledkem rovnice je hodnota BMR v jednotkách *kcal*. Například 50letá žena s váhou 65 kg a výškou 165 cm má BMR 1 341 kcal.

3.3.2 Mifflin-St Jeorova rovnice

Mifflin-St Jeorova rovnice je matematický vzorec, který se používá k výpočtu přibližné hodnoty BMR člověka. Rovnice byla vyvinuta v roce 1990 a pro výpočet využívá pohlaví, věk, výšku a hmotnost. Dle studie, která porovnávala výsledky a přesnost rovnic pro výpočet BMR, je Mifflin-St Jeorova nejpřesnější [7].

$$BMR\ ženy = (10 \times hmotnost\ kg) + (6{,}25 \times výška\ m) - (5 \times věk\ v\ letech) - 161 \qquad (4)$$

$$BMR\ muži = (10 \times hmotnost\ kg) + (6{,}25 \times výška\ cm) - (5 \times věk\ v\ letech) + 5 \qquad (5)$$

Výsledkem rovnice je hodnota BMR v jednotkách *kcal*. Například 50letá žena s váhou 65 kg a výškou 165 cm má BMR 1 270 kcal.

3.4 Energetický příjem a výdej

Energetický příjem je množství energie, kterou člověk získává z potravy a nápojů, které konzumuje. Energetický výdej je množství energie, které tělo spotřebuje za účelem provádění svých základních funkcí a různých fyzických aktivit. Je důležité, aby byl celkový energetický příjem a výdej v rovnováze, protože pokud je příjem vyšší než výdej, může dojít k nadváze a obezitě, zatímco pokud je výdej vyšší než příjem, může dojít k podvýživě. Rovnováha mezi příjmem a výdejem je tedy důležitá pro zdraví a dobrý fyzický stav.

3.4.1 Kalorický deficit

Kalorický deficit je situace, kdy celkový energetický příjem z potravy a nápojů je nižší než celkový energetický výdej během dne. To znamená, že člověk vynakládá více energie, než kolik přijímá, a tělo se musí spoléhat na rezervy energie uložené v tukových a svalových tkáních. Kalorický deficit je důležitý pro hubnutí. Pokud je ale příliš velký, může mít negativní vliv na zdraví [8]. Je důležité, aby byl kalorický deficit vyvážený a měl odpovídající výživové hodnoty, aby se zabránilo podvýživě.

3.4.2 Kalorický nadbytek

Kalorický nadbytek (surplus) nastává, když je celkový energetický příjem z potravy a nápojů vyšší než celkový energetický výdej během dne. To znamená, že člověk přijímá více energie, než kolik vynakládá, a tělo si ukládá přebytečnou energii formou tuku [9]. Kalorický nadbytek je důležitý pro přibírání na váze a budování

svalové hmoty, ale je třeba s ním zacházet s rozvahou, protože pokud je příliš velký, může to vést k nadváze, či dokonce obezitě a zhoršení zdravotního stavu.

3.4.3 Celkový energetický výdej

Celkový energetický výdej (TEE) je množství energie, kterou tělo vynakládá během dne na různé fyzické aktivity včetně bazálního metabolismu. Celkový energetický výdej je individuální v závislosti na pohlaví, věku, výšce, hmotnosti a úrovni fyzické aktivity. Hodnota TEE je výsledkem součinu bazálního metabolismu (Harris-Benedictova nebo Mifflin-St Jeorova rovnice) a konstanty dle úrovně fyzické aktivity [10] (Tabulka 2).

Tabulka 2 - Konstanty dle úrovně fyzické aktivity

Konstanta	**Úroveň fyzické aktivity**
1,2	Sedavý způsob života
1,375	Mírně aktivní
1,55	Středně aktivní
1,725	Velmi aktivní
1,9	Vysoce aktivní

3.5 Makroživiny

Makroživiny jsou základní živiny, které tělo potřebuje ke správnému fungování organismu. Rozdělujeme tři hlavní druhy makroživin: bílkoviny, sacharidy a tuky. Jsou považovány za esenciální živiny, což znamená, že si je tělo nedokáže samo vyrobit a je nutné je získávat z potravin. Je důležité, aby byl příjem makroživin vyvážený a aby odpovídal individuálním potřebám a cílům pro zajištění zdravého fyzického stavu [11].

3.5.1 Bílkoviny

Bílkoviny (proteiny) patří do kategorie makroživin. Jedná se o základní stavební složku buněk a tkání v těle. Bílkoviny jsou složeny z menších jednotek, které se

nazývají aminokyseliny. V těle se běžně nachází dvacet různých aminokyselin a každá z nich plní své důležité funkce. Celkem 9 z nich je esenciálních, tělo si je nedokáže vyrobit samo a je nutné je získávat z potravin. Aminokyseliny pomáhají v těle vytvářet nové bílkoviny, které jsou důležité pro růst a opravu tkání a budování svalů. Poskytují strukturu buněčným membránám, orgánům, vlasům, kůži a nehtům. Aminokyseliny se podílí na tvorbě enzymů, hormonů a správném fungování imunitního systému. Bílkoviny se nachází v masu, mléčných výrobcích, luštěninách, obilovinách a ořechách [12].

3.5.2 Sacharidy

Sacharidy jsou zdrojem rychle dostupné energie. Většina sacharidů se rozkládá na molekuly glukózy neboli cukru. Výjimkou je vláknina, což je specifický typ sacharidu, který se nerozkládá. Glukóza je varianta okamžité energie a je důležitá pro správné fungování mozku, nervového systému a červených krvinek. Přebytečná glukóza se ukládá ve formě glykogenu ve svalech a játrech, pro pozdější použití. Sacharidy se nachází především v obilovinách, luštěninách, ovoci a zelenině [13].

3.5.3 Tuky

Tuky se v těle štěpí na mastné kyseliny a glycerol. Jsou zdrojem energie, pomáhají udržovat teplotu těla, ochraňují orgány a jsou základní složkou buněčných membrán. Přenášejí a napomáhají vstřebání důležitých látek, jako například vitamíny A, D, E a K. Nachází se v mase, mléčných výrobcích, ořechách a olejích [14].

4 Současné aplikace na trhu

Sledování kalorií a výživových hodnot potravin je běžný způsob, jak mohou lidé dosáhnout svých cílů v oblasti tělesné váhy a výživy. Mnoho aplikací a webových stránek je pro tyto účely k dispozici a mnoho lidí je používá jako nástroj pro správné stravování a dodržování zdravého životního stylu. Aplikace poskytují uživatelům analýzy a přehled o jejich pokroku a pomáhají jim provádět úpravy ve stravě pro dosažení jejich cílů. Jedná se o užitečné nástroje pro lidi, kteří se snaží zhubnout, udržet zdravou váhu nebo se jen chtějí dozvědět více o svých stravovacích návycích. Zvláště užitečné jsou pro lidi, kteří mají potíže sledovat svůj příjem kalorií nebo potřebují extra motivaci a podporu.

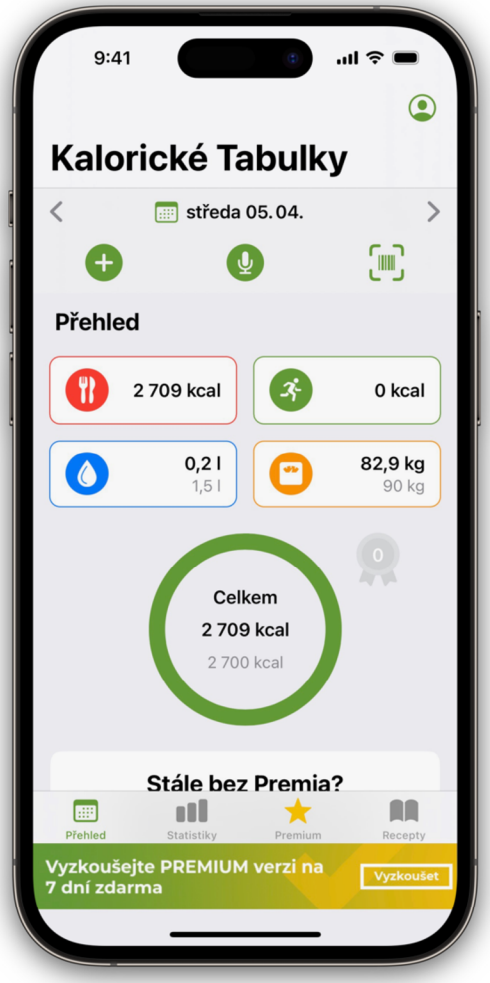
4.1 Kalorické tabulky

Kalorické tabulky je mobilní a webová aplikace, kterou provozuje firma Dine4Fit. Aplikace nabízí rozsáhlou databázi, která poskytuje uživatelům informace o kalorickém a výživovém obsahu potravin (Obrázek 1). Cílem aplikace je pomoci uživatelům udržet si přehled o přijatých kaloriích, aby mohli lépe sledovat příjem potravy a dosáhnout svých zdravotních cílů. Kalorické tabulky evidují šest milionů registrovaných uživatelů a zhruba 200 tisíc denně aktivních uživatelů [15].

Jednou z mnoha funkcí je možnost zaznamenávání tělesné aktivity, čímž se automaticky upravuje příjem kalorií. Dále osahuje i komunitní funkce, díky kterým mohou uživatelé sdílet své výsledky a rady s ostatními.

Základní bezplatná verze aplikace umožňuje vyhledávat potraviny v databázi dle názvu nebo dle čárového kódu a zapisovat je do jídelníčku. Následně zobrazuje grafy a statistiky o aktuálním kalorickém příjmu a hodnoty sacharidů, bílkovin a tuků [16].

Kromě základní bezplatné verze je k dispozici také placená prémiová verze. Ta disponuje dalšími funkcemi, jako například sledování aditivních potravinářských látek (tzv. éčka), přístup ke vzorovým jídelníčkům, rozšíření grafů a statistik o hodnoty cukru, soli, vápníku a mastných kyselin. V této verzi aplikace nejsou reklamy [17].



Obrázek 1 - Aplikace Kalorické Tabulky (vlastní zpracování)

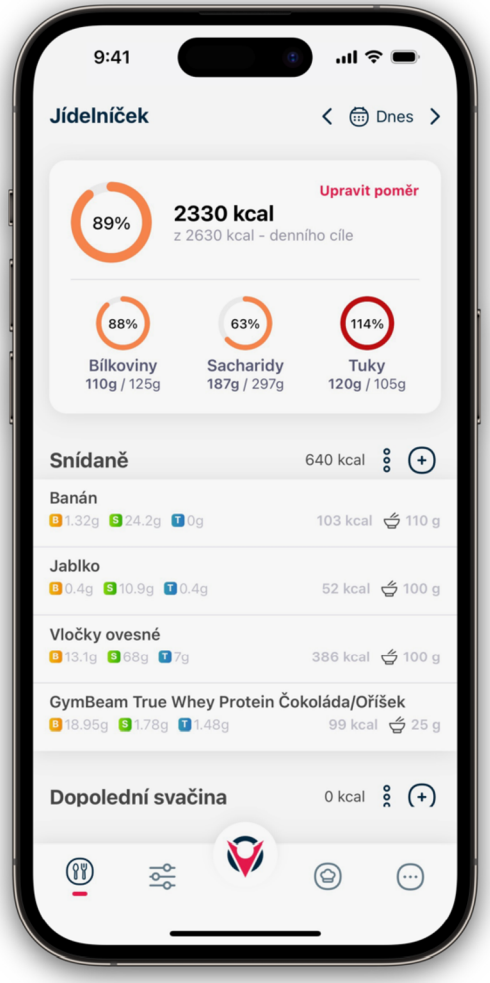
4.2 Macronaut

Macronaut je aplikace, kterou provozuje stejnojmenná česká firma, pro sledování kalorií a výživových hodnot potravin (Obrázek 2). Umožňuje uživatelům zaznamenávat svůj denní příjem potravy a cvičení, a poskytuje jim informace o tom, kolik kalorií, bílkovin, tuků a sacharidů konzumují [18].

Aplikace obsahuje rozsáhlou databázi potravin a má funkci pro vyhledávání podle názvu nebo čárového kódu. Umožňuje uživatelům nastavit cíle pro příjem kalorií, bílkovin, tuků a sacharidů, a sledovat, jak se jim daří naplňovat stanovené cíle [18]. Aplikace obsahuje grafy a statistiky, díky kterým uživatelé vidí, jak se jejich

stravovací návyky mění, také umožňuje vytvoření jídelníčku a plánování jídla dopředu.

Na rozdíl od Kalorických tabulek nenabízí Macronaut webovou verzi a uživatel je tak odkázán pouze na mobilní aplikaci. Hlavním rozdílem však pro řadu uživatelů je absence bezplatné varianty. Macronaut nabízí pouze sedmi denní zkušební období a následně je nutné si aplikaci předplácet [19].



Obrázek 2 - Aplikace Macronaut (vlastní zpracování)

4.3 Lose It!

Lose It! je mobilní aplikace, která pomáhá uživatelům sledovat příjem kalorií, cvičení a celkové zdravotní cíle (Obrázek 3). Je vyvíjena americkou společností FitNow Inc. Aplikace umožňuje nastavit personalizované cíle a sledovat pokrok v čase [20]. Jednou z hlavních funkcí aplikace Lose It! je její rozsáhlá databáze potravin, která obsahuje informace o kalorickém a výživovém obsahu více než 7 milionů potravin a nápojů včetně těch na českém trhu. Aplikace také obsahuje nástroje pro sledování makroživin a příjmu tekutin [20].

Obrázek 3 - Aplikace LoseIt! (vlastní zpracování)

5 Neuronové sítě

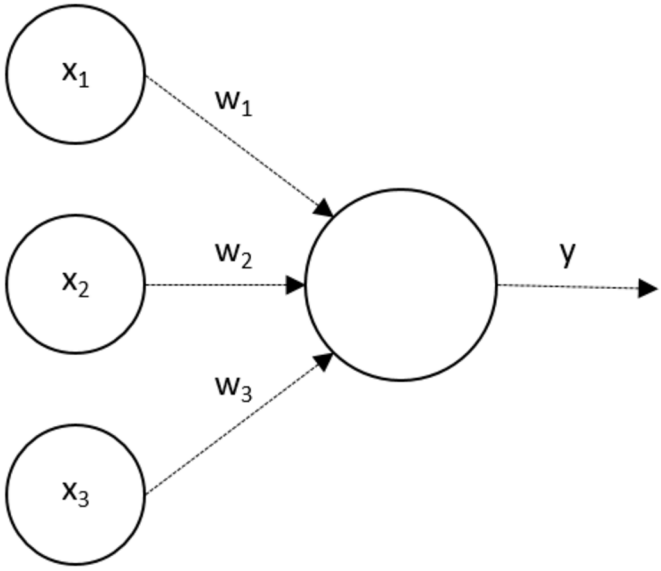
Umělá neuronová síť je matematický model používaný v oboru umělé inteligence, který je inspirován principy fungování lidského mozku a nervového systému [21]. Pracuje tak, že přijímá vstupní signály, které zpracovává a vytváří výstupní signál. Umělé neuronové sítě vycházejí z biologické analogie s nervovým systémem, který přijímá přicházející signály ze smyslových receptorů (vstup). Ty jsou zpracovány, vyhodnoceny a je vyslán pokyn k efektorům (výstup), čímž je zajištěna odpovídající reakce, například v podobě pohybu svalu. Fungování systému si lze představit při hodu míčem na cíl. Pokud při prvním pokusu mineme, nervový systém vyhodnotí signály z receptorů (například vizuální vjem) a hodnoty porovná s požadovaným stavem (trefa míče na cíl). Výsledkem jsou upravené hodnoty (například větší nebo menší zapojení svalů) pro nový pokus hodu míčem [22]. Prostřednictvím zpětné vazby umělá neuronová síť, potažmo neuronový systém, zdokonaluje své schopnosti, díky možnosti pamatovat si informace na základě předchozích stavů, z nich se učit a tím dosáhnout lepších výsledků [23].

S určitou formou umělé inteligence se lze v dnešním světě setkat poměrně snadno. Neuronové sítě se používají pro různé úkoly, jako je klasifikace fotografií, rozpoznávání řeči nebo predikce budoucích hodnot. Mobilní telefony využívají neuronové sítě při diktování pro převod hlasu na text nebo chytří asistenti jako například Siri, Alexa nebo Google Assistant pro zpracování jazyka a porozumění textu. Mnohé sociální sítě, jako například Facebook využívají pokročilé funkce rozpoznávání obličeje pro detekci uživatelů na fotografiích nebo analýzu osobních dat a chování uživatele pro personalizaci nabízených příspěvků a cílených reklam. Další uplatnění umělých neuronových sítí lze nalézt ve finanční sféře, marketingu, zdravotnictví, herním nebo automobilovém průmyslu.

5.1 Umělý neuron

Umělý neuron je matematický model, který se inspiroval principy fungování biologického neuronu (například v nervové soustavě). Umělý neuron přijímá vstupy, které jsou určitým způsobem zpracovány a generují výstup, který je

přenášen do dalších umělých neuronů formou vstupní informace. Zpracování vstupů se provádí pomocí různých matematických operací, jako jsou lineární transformace nebo aktivační funkce. Vstupy a výstupy jsou reprezentovány jako vektory nebo matice [24]. V neuronové síti se jednotlivé umělé neurony spojují mezi sebou váhami, které reprezentují sílu vztahu mezi vstupy a výstupy jednotlivých neuronů [25]. Umělý neuron je základní stavební jednotka neuronové sítě, která se skládá z mnoha těchto umělých neuronů, vzájemně propojených a organizovaných ve vrstvách.



Obrázek 4 - Model umělého neuronu [23]

V modelu umělého neuronu (Obrázek 4) jsou vstupy (x_n) vynásobeny váhami (w_n) a zpracovány v těle umělého neuronu, kde vznikne suma těchto dílčích součinů. Výsledkem matematické operace je vnitřní potenciál neuronu (y). V případě, že potenciál přesáhne stanovenou prahovou hodnotu (bias), je dále zpracován aktivační funkcí a odeslán na výstup neuronu [23; 26]. Z pohledu matematiky popisuje umělý neuron rovnice:

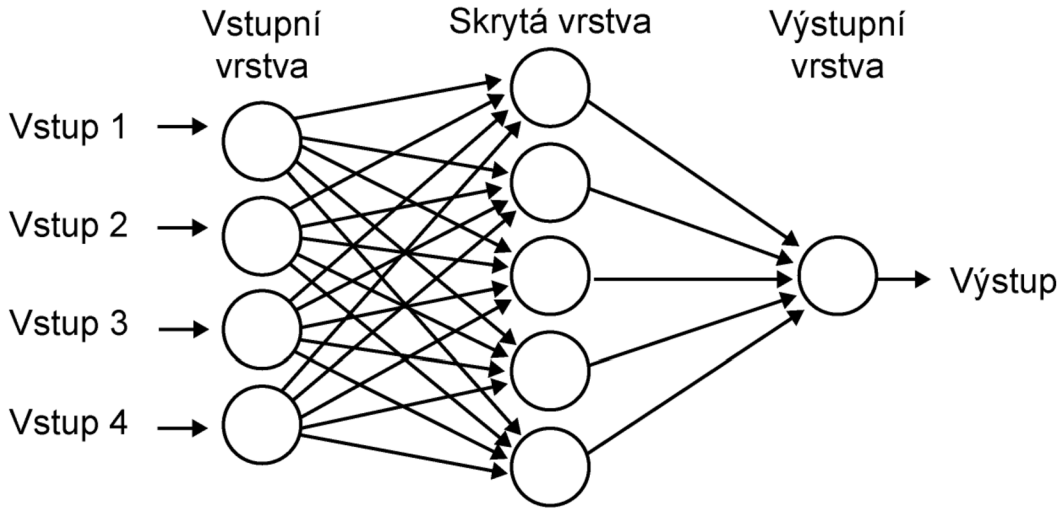
$$y = f\left(\sum_{i=1}^{n}(x_i\, w_i) - b\right) \qquad (6)$$

Pro rovnici platí následující vtahy:

- `y` je výstup neuronu, který může být použit jako vstup pro další vrstvy.
- `f` je aktivační funkce, která zpracovává vnitřní potenciál neuronu.
- `b` je prahová hodnota aktivační funkce. Pokud je hodnota $\sum_{i=1}^{n}(x_i\, w_i)$ větší než práh, neuron je aktivní. V opačném případě zůstává neaktivní.
- `n` je počet vstupů do neuronu.
- `x` je vstupní hodnota neuronu.
- `w` je váha, která reprezentuje sílu vztahu mezi vstupy a výstupy. Čím je hodnota vyšším, tím je daný vstup důležitější.

5.2 Umělá neuronová síť

Umělá neuronová síť je tvořena matematickými (umělými) neurony, které jsou vzájemně propojeny a uspořádány ve vrstvách. Platí pravidlo, že neexistuje propojení mezi neurony jedné vrstvy, avšak mezi neurony sousedních vrstev existuje propojení úplné. První vrstva umělé neuronové sítě se nazývá vstupní a poslední vrstva je označována jako výstupní. Ostatní vrstvy, které se nachází mezi dvěma zmíněnými, se nazývají skryté vrstvy [27] (Obrázek 5).

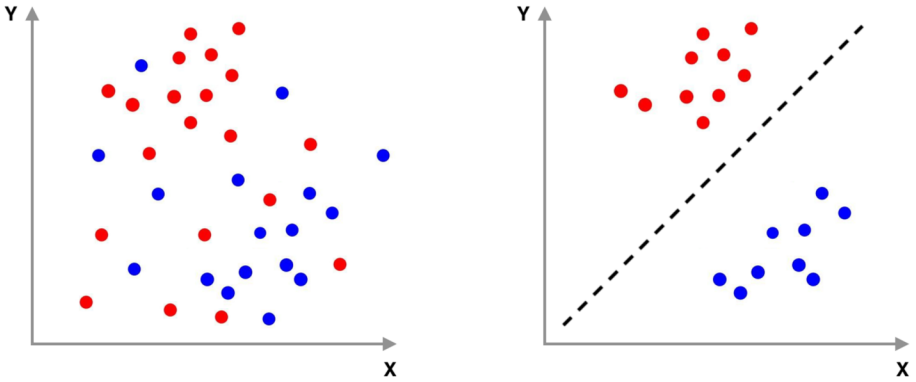


Obrázek 5 - Uspořádání vrstev a neuronů v umělé neuronové síti [27]

Spoje mezi neurony představují cesty pro šíření signálů, jsou orientovány a každý spoj je ohodnocen váhou, která modifikuje intenzitu procházejícího signálu. To, jakým způsobem umělá neuronová síť převádí vstupní data na požadovaný výstup, se určuje v procesu zvaném učení. Cílem učení je stanovit takové hodnoty vah a prahů, aby se hodnota výstupu sítě co nejvíce rovnala požadovanému výsledku. Pokud se nerovnají, síť provede korekci vah a prahů a celý proces se opakuje do té doby, než je iteracemi docíleno požadované hodnoty na výstupu. Proces učení lze rozdělit na dvě hlavní kategorie – učení s učitelem (supervised learning) a učení bez učitele (unsupervised learning) [27].

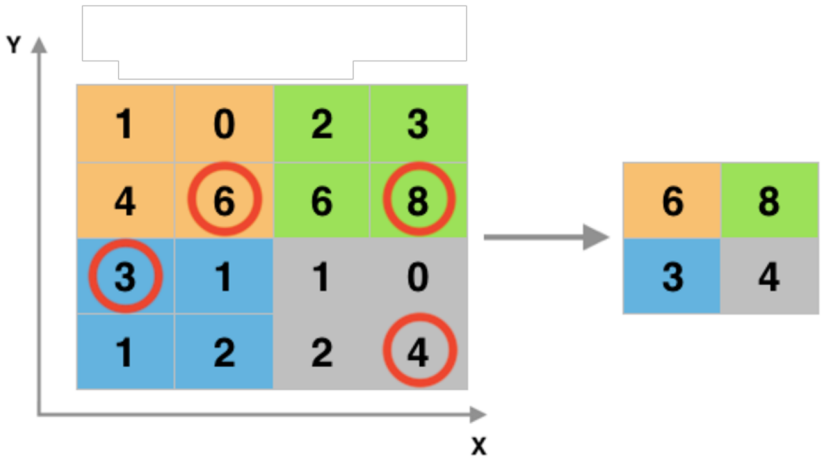
Učení s učitelem je založeno na znalosti správných výstupů, které jsou umělé neuronové sítí předloženy společně se vstupními daty (například množina obrázků, které jsou kategoricky roztříděny do tříd). Během procesu učení síť nastavuje vnitřní hodnoty vah a prahů tak, aby se výstup co nejvíce rovnal požadované předem známé hodnotě [23].

Varianta učení bez učitele nezahrnuje znalost správných výstupů. To znamená, že v tomto případě musí síť sama najít vzorce a vztahy mezi vstupními daty. Během procesu se obvykle provádí shluková analýza neboli roztřídění vstupů podle vzorců a vztahů (Obrázek 6). To umožňuje rozdělit data do menších skupin na základě podobností nebo společných vlastností [23].



Obrázek 6 – Rozdělení prostoru dat po shlukové analýze (vlastní zpracování)

Umělé neuronové sítě se často zabývají zpracováním fotografií. Při práci s obrazovými daty je problém jejich velikost. V případě fotografie 100×100 pixelů se třemi barevnými kanály (RGB) se jedná o vstupní data o velikosti 30 000 hodnot. To představuje pro neuronovou síť problém v podobě velmi velkého množství parametrů, které vyžadují vysoký výpočetní výkon na své zpracování. Z tohoto důvodu se využívají takzvané konvoluční neuronové sítě, které jsou navržené tak, aby zpracovávaly fotografii postupně po malých částech, například 3×3. Taková síť by měla na svém vstupu pouze 27 parametrů, a to představuje znatelný rozdíl oproti původní hodnotě 30 000. Konvoluce je matematická operace, která se používá k rozpoznání vzorců v datech. Konvoluční síť se skládá z několika takových konvolučních vrstev a každá obsahuje několik filtrů. Ty se postupně posouvají přes vstupní data a provádějí konvoluci skrze filtry, které detekují rysy jako například hrany, rohy, kruhy a další tvary. Součástí vrstev v konvoluční síti jsou takzvané pooling vrstvy. Ty se aplikují stejně jako konvoluční vrstvy po částech o malých velikostech, například 2×2. Úkolem pooling vrstvy je rozdělení fotografie na části a výběru jedné hodnoty (typicky maximální) z dané oblasti. Tím je docíleno zmenšení velikosti obrazových dat. V konvolučních sítích se zpravidla střídají konvoluční a pooling vrstvy. Následující příklad demonstruje pooling vrstvu o velikosti 2×2, která zmenšila původní obrazová data o 75 % výběrem maximální hodnoty z oblasti (Obrázek 7) [28; 29].



Obrázek 7 - Ukázka pooling vrstvy [28]

6 Softwarové technologie

Následující kapitola představuje důležité technologie, programovací jazyky, knihovny, frameworky a softwarové nástroje, které byly použity při vývoji aplikace.

6.1 Swift & SwiftUI

Swift je programovací jazyk vyvinutý společností Apple a používá se k vývoji aplikací pro operační systémy macOS, iOS, iPadOS, watchOS a tvOS. Swift byl představen v roce 2014 a postupně nahrazuje jazyk Objective-C, který se do té doby pro vývoj aplikací pro zmíněné operační systémy využíval. Swift je navržen tak, aby byl bezpečnější, efektivnější a snadnější na použití než Objective-C. Podporuje moderní programovací paradigma, jako je objektově orientované nebo funkcionální programování a programování reaktivních aplikací. Jednou z hlavních vlastností jazyka Swift je jeho čitelnost a srozumitelnost, což umožňuje vývojářům psát udržitelnější kód. Další vlastností je jeho vysoká úroveň bezpečnosti a stability, díky čemuž se snižuje riziko výskytu chyb a zároveň se zlepšuje výkon aplikací [30].

Součástí vývojových nástrojů společnosti Apple je také framework SwiftUI, který se používá pro vývoj moderních uživatelských rozhraní v jazyce Swift. Použitelnost SwiftUI spočívá v tom, že poskytuje (znovu)použitelné komponenty (tlačítka, vstupní pole, interaktivní prvky, pohledy), které se díky podpoře adaptivního rozvržení dokážou přizpůsobit různým platformám, rozlišením obrazovek a orientací zařízení. Dále umožňuje vytvářet dynamická a interaktivní rozhraní pomocí jednoduché syntaxe a automaticky synchronizuje změny uživatelského rozhraní s modely dat.

SwiftUI přináší nový způsob vytváření uživatelského rozhraní, který se zaměřuje na zjednodušení práce s kódem a usnadnění vývoje aplikací [31].

6.2 React.js

React.js je open-source framework a JavaScriptová knihovna vyvinutá společností Facebook. Používá se k rychlému a efektivnímu vytváření webových aplikací

a interaktivních uživatelských rozhraní. Framework React.js kombinuje rychlost a efektivitu jazyka JavaScript, což umožňuje rychlejší vykreslování webových stránek a vytváření dynamických a responzivních webových aplikací [32].

Aplikace v React.js jsou vyvíjeny tvorbou (znovu)použitelných komponent, které si lze představit jako nezávislé bloky, které po sestavení tvoří celé uživatelské rozhraní aplikace [32]. Komponenty mají své vlastnosti formou vstupních parametrů (props) a spravují svůj vnitřní stav (state). Tento deklarativní způsob práce s daty aplikace vede k více předvídatelnému chování i snadnějšímu ladění [33].

6.3 Next.js

Next.js je JavaScriptový framework pro vytváření webových aplikací. Je postaven na React.js, populární knihovně JavaScriptu pro tvorbu interaktivních uživatelských rozhraní. Next.js poskytuje sadu funkcí a nástrojů pro vytváření server-renderovaných React aplikací. To znamená, že Next.js aplikace před-renderují HTML na serveru a posílají ho klientovi, což umožňuje poskytovat obsah stránek uživatelům rychleji. To je užitečné pro zlepšení výkonu aplikací na pomalém internetovém připojení nebo na zařízeních s nízkými specifikacemi [34].

Další klíčovou funkcí Next.js je automatické dělení kódu. Umožňuje rozdělit kód na menší části, které jsou na požádání načteny. Tím se zlepšuje výkon aplikace snížením množství kódu, který je nutné načíst. Next.js také poskytuje vestavěný vývojový server a podporuje funkci Hot module reloading, což umožňuje vidět změny v kódu ihned bez nutnosti obnovit stránku [34].

Jedná se o silný, výkonný a efektivní framework pro tvorbu moderních webových aplikací.

6.4 TypeScript

TypeScript je silně typovaný, objektově orientovaný, kompilovaný programovací jazyk, který vychází z jazyka JavaScript. Jedná se o nadstavbu jazyka JavaScript, která je navržena tak, aby poskytovala statickou typovou kontrolu, což znamená,

že umožňuje definovat typy proměnných, funkcí, tříd a dalších konstrukcí. To vede ke snížení počtu chyb v kódu a zlepšení jeho čitelnosti.

Hlavním architektem jazyka TypeScript je Anders Hejlsberg, designér jazyka C# ze společnosti Microsoft. TypeScript dále poskytuje funkce jako rozhraní, třídy, dědičnost a generické typy, které jsou užitečné pro větší a komplexnější projekty. TypeScript je kompatibilní s existujícími knihovnami a frameworky pro JavaScript, jako například React, Angular nebo Node.js.

TypeScript může být kompilován do JavaScriptu, tudíž kód naprogramovaný v TypeScriptu lze spustit na jakémkoli prostředí, které podporuje JavaScript [35].

6.5 API & REST

API (Application Programming Interface) je rozhraní, které umožňuje vzájemnou komunikaci mezi různými systémy nebo aplikacemi. Soubor definic a protokolů určuje, jakým způsobem probíhá komunikace a vyměňování informací. API poskytuje množinu funkcí, metod, datových typů a protokolů, které umožňují vytvářet interakce mezi aplikacemi a systémy.

API je často implementováno pomocí REST nebo GraphQL protokolu a poskytují přístup k datům nebo funkcím webové aplikace pomocí standardních HTTP požadavků. API se nejčastěji používá pro přenos dat mezi webovými, mobilními a serverovými aplikacemi [36].

REST (Representational State Transfer) je architektonický styl pro návrh distribuovaných systémů a webových služeb. Je založen na protokolu HTTP a definuje způsob, jakým klient a server komunikují. Přístup ke zdrojům je řízen standardními metodami protokolu HTTP, jako GET, POST, PUT a DELETE. Jednotlivé zdroje jsou na straně serveru identifikovány unikátními URL adresami. REST aplikace jsou bez stavu, což znamená, že veškeré informace, potřebné pro zpracování požadavku, jsou k dispozici pouze během relace a po jejím ukončení nejsou data na serveru uchována. REST se často používá pro vývoj webových služeb, protože se jedná o flexibilní způsob, jak poskytnout přístup k datům přes síť.

REST API jsou snadno implementovatelné a použitelné pro různé jazyky a platformy [37].

6.6 CoreML & CreateML

CoreML (Core Machine Learning) je framework pro strojové učení vyvinutý společností Apple a lze jej využít na platformách iOS a macOS. CoreML umožňuje vývojářům přenášet modely strojového učení do aplikací a využívat modely strojového učení vytvořené v jiných frameworcích jako TensorFlow. CoreML umožňuje vývojářům efektivně implementovat strojové učení a vytvářet inteligentní aplikace, které se mohou učit a přizpůsobovat uživatelům [38].

Předností frameworku CoreML je, že modely strojové učení mohou být spuštěny přímo na zařízení, nezávisle na připojení k internetu. To je důležité pro aplikace s vysokým důrazem na soukromí uživatelských dat, která zůstávají na zařízení nebo pro aplikace využívající strojové učení v místech bez internetového připojení.

CreateML je nástroj vytvořený společností Apple pro tvorbu modelů strojového učení. Umožňuje vývojářům vytvářet a trénovat modely strojového učení pomocí grafického rozhraní bez nutnosti větší znalosti programování. CreateML nabízí několik předpřipravených modelů pro regresní analýzu, detekci objektů, klasifikaci nebo segmentaci obrazu.

Modely CoreML vytvořené prostřednictvím nástroje CreateML jsou velmi dobře optimalizované pro mobilní zařízení. Využívají vysoký výkon při malé paměťové stopě a nízkému množství spotřebované energie díky speciálním akceleračním čipům Neural Engine, které jsou součástí většiny produktů společnosti Apple.

6.7 PostgreSQL

PostgreSQL je relační databázový systém, který poskytuje pokročilé funkce pro správu dat. Databáze je systém pro ukládání, organizaci a vyhledávání dat, jako jsou textové informace, čísla, obrázky nebo multimediální obsah. Data jsou ukládána v tabulkách, souborech nebo jiných organizovaných strukturách.

Databáze slouží jako zdroj informací pro aplikace a systémy, které potřebují pracovat s daty. Aplikace a systémy mohou přistupovat k datům v databázi pomocí dotazů a dalších operací, což jim umožňuje číst, zapisovat, aktualizovat a mazat data v databázi.

PostgreSQL je jeden z nejvyspělejších a nejvýkonnějších databázových systémů na světě. Nabízí řadu funkcí, které umožňují efektivně spravovat a organizovat data, například podpora pro triggery, uložené procedury, indexování, transakční zpracování, fulltextové vyhledávání nebo práci s geoprostorovými daty. Díky tomu je možné efektivně pracovat s velkými objemy informací [39].

7 Návrh aplikace

K implementaci aplikace byla zvolena architektura klient-server. Jedná se o typ architektury aplikací, kde existuje jeden nebo více serverů, které poskytují služby nebo zdroje klientům. Klienti jsou počítače nebo mobilní zařízení, které tyto služby a zdroje využívají. Klienti se připojují k serverům a odesílají žádosti, které server přijímá a zpracovává. Server poté vrací odpověď klientovi. Tento model komunikace je často používán pro aplikace, které vyžadují sdílení dat nebo služeb mezi mnoha uživateli nebo kde je potřeba centralizovat data nebo služby [40].

Klient a server jsou oddělené entity, které komunikují pomocí internetových protokolů, jako například HTTP. Klient zobrazuje data a umožňuje uživateli interakci s nimi, zatímco server data poskytuje a provádí nad nimi operace.

Architektura klient-server se často používá pro vytvoření distribuovaných systémů, jako jsou webové aplikace a e-mailové nebo databázové systémy. Tyto systémy se skládají z mnoha klientů, kteří se připojují k jednomu nebo více serverům, které poskytují data a služby. V případě webových aplikací lze webový prohlížeč považovat za klient komunikující se vzdáleným webových serverem, který poskytuje data [41].

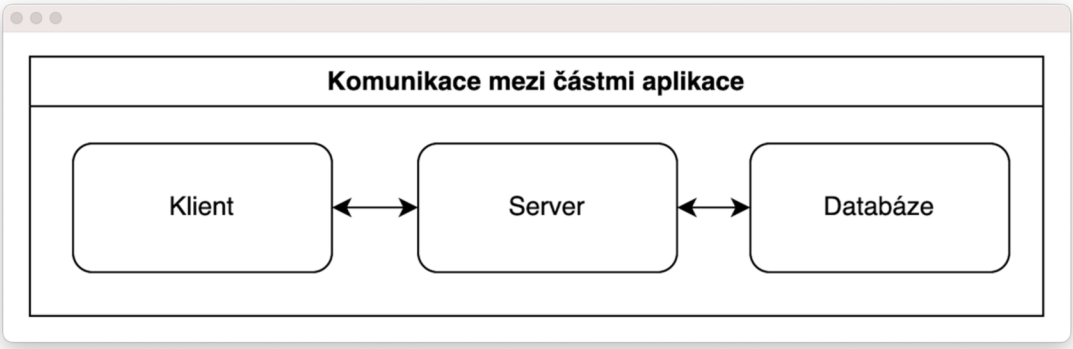
Služby poskytované serverem jsou centralizovány, což umožňuje efektivnější správu a údržbu. Data, uložená na serveru, mohou být lépe chráněna před neoprávněným přístupem a v případě nedostatku hardwarových zdrojů lze kapacitu serveru navýšit [40].

Samotná komunikace a přenos dat mezi klientem a serverem využívá rozhraní API a architektonický styl pro navrhování aplikací REST. Kombinace těchto dvou technologií se označuje jako RESTful API [42].

7.1 Server

Serverová část aplikace je založena na architektuře webových služeb. Jedná se o jednotný systém, který umožňuje vzájemnou komunikaci s jinými aplikacemi, nezávisle na programovacím jazyce nebo platformě. Server je implementován JavaScriptovým frameworkem pro tvorbu webových aplikací Next.js. Zvolené řešení poskytuje klientovi API pomocí definovaných URL adres, které umožňují snadnou a efektivní komunikaci mezi mobilní aplikací a serverem. API implementuje standardní metody CRUD (create, retrive, upate, delete) pro práci s daty v databázi. Tato architektura zajišťuje nejen hladký chod aplikace, ale i bezpečné a spolehlivé zpracování uživatelských dat.

Veškerá data jsou ukládána do relačního databázového systému PostgreSQL a o jejich správu se stará výhradně server. Samotný klient nikdy nepřistupuje k databázi napřímo, ale pouze skrze prostředníka, kterým je server (Obrázek 8).



Obrázek 8 - Schéma komunikace mezi částmi aplikace (vlastní zpracování)

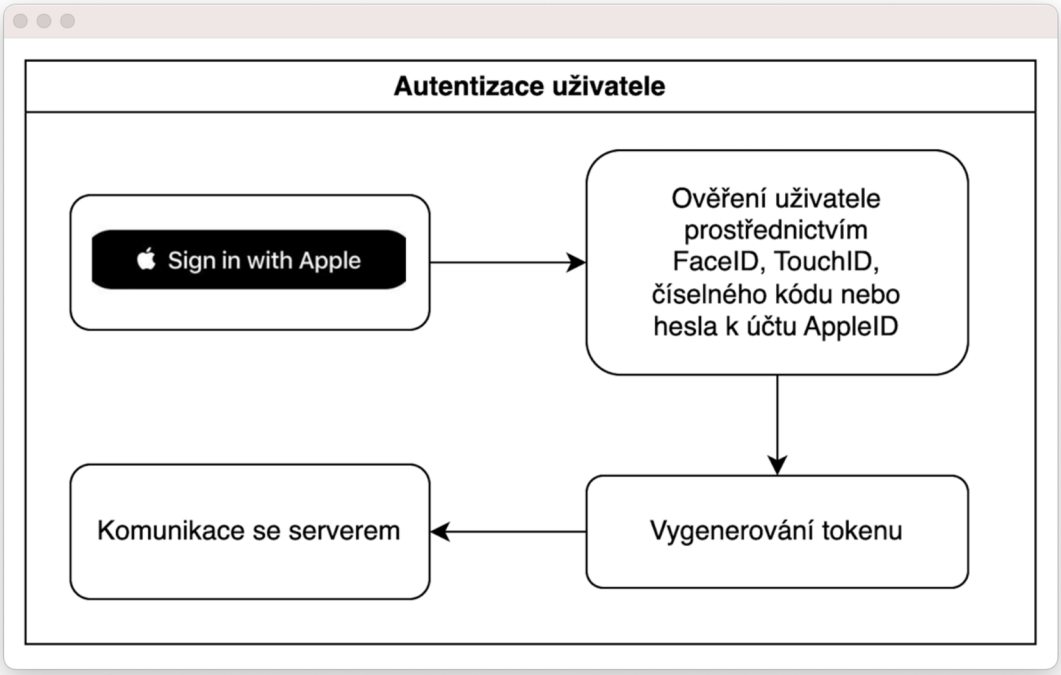
Komunikace mezi klientem a serverem probíhá prostřednictvím zabezpečeného šifrovaného připojení díky protokolu HTTPS a každý požadavek na serveru podléhá ověření autentizačního tokenu, který je unikátní pro každého uživatele. V případě že token není k dispozici nebo je neplatný, požadavek není obsloužen, protože není možné ověřit autentizaci uživatele.

7.2 Klient

Klientem se rozumí mobilní aplikace vyvinutá v jazyce Swift na platformě iOS. Klient poskytuje uživatelům přehledné uživatelské rozhraní, které využívá framework SwiftUI.

Aplikace skrze server přistupuje k databázi potravin s informacemi o kalorickém obsahu a výživových hodnotách. Uživatelé mohou vyhledávat potraviny, přidávat je do svého jídelníčku a sledovat svůj příjem kalorií a makroživin. Aplikace také umožňuje skenování čárových kódů a rozpoznání potravin prostřednictvím kamery na zařízení. Novým uživatelům poskytuje doporučení pro příjem kalorií a makroživin na základě vyplnění vstupního dotazníku. Dále jsou k dispozici statistiky a personalizovaná doporučení pro uživatele na základě jejich stravovacích návyků.

Uživatelé jsou autentizováni pomocí služby Sign in With Apple. Ta umožňuje uživatelům rychlé a bezpečné přihlášení do aplikací pomocí svého AppleID, místo toho, aby museli vyplňovat registrační formulář a zadávat osobní informace, jako například jméno, e-mailová adresa a heslo. Než může uživatel využívat funkci Sign in With Apple, musí být na svém zařízení přihlášený pod svým AppleID, a to včetně dvoufaktorového ověření (Obrázek 9). Tento proces zvyšuje bezpečnost účtu a snižuje riziko neoprávněného přístupu. Služba dále uživatelům umožňuje skrýt svoji skutečnou e-mailovou adresu. Místo toho Apple vygeneruje unikátní anonymní e-mailovou adresu, která je poskytnuta aplikaci. Služba se dále postará o přeposílání zpráv na skutečnou adresu. Tímto způsobem mohou uživatelé zvýšit své soukromí.



Obrázek 9 - Schéma přihlášení do aplikace (vlastní zpracování)

Klient komunikuje se serverem skrze API, které definuje přístupové URL adresy pro výměnu dat. Součástí všech požadavků je autentizační token, který je vygenerován po přihlášení uživatele. Server ověří platnost tokenu a po jeho dekódování získá informace o konkrétním uživateli. Token může být neplatný, pokud již vypršela jeho platnost. V tom případě je uživatel aplikace vyzván k opětovnému přihlášení. Další možností je, že došlo k neoprávněnému zásahu a změně obsahu tokenu před jeho odesláním, čímž se stává neplatným a pravděpodobně se jedná o podvrh.

Pro uživatele přináší služba Sign in With Apple výhody související se snadným ovládáním, soukromím a bezpečností. Vývojáři aplikací se nemusí zabývat ukládáním citlivých údajů uživatele, jako například heslo, protože jsou zpravovány na straně služby.

7.3 Funkční požadavky

Vyvíjená aplikace by měla splňovat následující klíčové funkční požadavky.

7.3.1 Databáze potravin

Aplikace disponuje vlastní databází jídel, potravin a nápojů, která obsahuje informace o kalorických a výživových hodnotách potravin. To uživatelům umožňuje vyhledávat jídla, přidávat je do svého jídelníčku a následně sledovat, kolik kalorií a živin konzumují. Díky tomu je možné vypočítat celkový příjem kalorií a makroživin za určité časové období pro každého uživatele.

Databáze potravin obsahuje zhruba 10 000 záznamů, které byly vytvořeny autorem diplomové práce a částečně převzaty v souladu s autorským právem a licenčními podmínkami pro užití z dostupných otevřených zdrojů OpenFoodFacts.org [43] a NutriDatabaze.cz [44]. Převzaté záznamy jsou použity pouze pro účely výzkumu této diplomové práce a nejsou využívány komerčně nebo distribuovány třetím stranám.

7.3.2 Sledování kalorického příjmu

Uživatelé mají možnost přesně zaznamenat množství kalorií, které každý den konzumují, což jim umožní přesněji plánovat svůj stravovací režim, sledovat svůj pokrok v rámci splnění stanovených zdravotních cílů a v neposlední řadě nabýt povědomosti o svém jídelníčku, stravovacích návycích a lépe plánovat stravovací režim.

7.3.3 Vyhledávání potravin

Uživatelé mohou procházet databázi potravin prostřednictvím funkce vyhledávání. Klíčovým parametrem pro vyhledávání je název potraviny, který slouží jako základ pro vyhledávací algoritmus. Ten prohledá databázi a nejde všechny potraviny, jejichž název se shoduje nebo je dostatečně podobný zadanému klíčovému slovu. Výsledky vyhledávání jsou uživatelům zobrazeny ve formě seznamu, ze kterého mohou následně vybrat cílenou potravinu pro zápis do jídelníčku.

7.3.4 Skenování čárových kódů potravin

Manuální vyhledávání podle názvu potravin může být pro uživatele nekomfortní a časově náročné. Alternativou je skenování čárových kódů, což může zlepšit přesnost a snížit chybovost při zapisování potravin do jídelníčku. Aplikace podle hodnoty čárového kódu vyhledá příslušnou potravinu v databázi a získá přesné informace o kalorickém obsahu a výživových hodnotách.

7.3.5 Rozpoznání potraviny z fotografie

Další alternativou k manuálnímu zapisování potravin do jídelníčku je rozpoznání potraviny z fotografie. Uživatelé mohou pořídit fotografii potraviny, aplikace ji rozpozná a poskytne řadu možných kandidátů, ze kterých uživatelé vyberou požadovanou potravinu. Tento proces zápisu potravin do jídelníčku může být pro uživatele přívětivější než manuální vyhledávání, protože nezahrnuje takové množství interakce, a díky tomu může být ve výsledku rychlejší a dlouhodobě motivující z důvodu snadnějšího postupu celkovým procesem zápisu potravin.

7.3.6 Výpočet doporučeného příjmu kalorií a makroživin

Aplikace poskytne uživateli individuální hodnoty doporučeného denního příjmu kalorií a makroživin. To umožňuje uživatelům získat přesné doporučení pro kalorický příjem v závislosti na pohlaví, věku, tělesné hmotnosti, výšce, úrovni fyzické aktivity a stanoveném cíli. Pokud člověk konzumuje více kalorií, než spotřebuje, může přibírat na váze a časem se objeví zdravotní problémy spojené s obezitou. Naopak, pokud člověk konzumuje méně kalorií, než kolik potřebuje, může trpět nedostatkem živin a energie. Správné nastavení úrovně kalorického příjmu je klíčové pro udržení zdravého životního stylu.

Pro výpočet hodnoty bazálního metabolismu je využita Mifflin-St. Jeorova rovnice. Výsledná hodnota představuje množství energie, které člověk potřebuje k udržení základních tělesných funkcí v klidovém stavu. Pro získání doporučeného denního příjmu kalorií je nutné výsledek vynásobit patřičným koeficientem, který zohlední úroveň fyzické aktivity člověka.

Doporučený příjem makroživin se vypočítá na základě předchozího výsledku. Obecně platí že příjem sacharidů by měl tvořit 45–65 %, tuků 20–35 % a bílkovin 10–35 % celkového doporučeného příjmu kalorií [45].

7.3.7 Statistiky a doporučení

Díky nashromážděným údajům může aplikace nabídnout řadu užitečných statistik a poskytnout uživateli lepší kontrolu nad svým zdravím a kondicí. Statistiky mohou identifikovat oblasti, ve kterých mají uživatelé problémy a navrhnout zlepšení těchto oblastí pro úspěšné dosažení stanovených cílů. Pokud ze statistik uživatelé vidí, že přijímají příliš mnoho kalorií nebo že nejsou dostatečně aktivní, mohou se rozhodnout pro úpravu svých stravovacích návyků. Pro uživatele je užitečné vidět, jaký pokrok udělali a jak se zlepšili v průběhu času, a to může být motivující pro pokračování v jejich snaze dosáhnout dlouhodobého zdraví a kondice.

Aplikace by měla nabízet personalizovaná doporučení na základě údajů o uživateli a jeho stravovacích návycích. Může navrhovat potraviny s vyšším obsahem určitých makroživin, pokud dle jídelníčku identifikuje jejich nedostatek. V případě, že uživatel konzumuje příliš mnoho potravin, které negativně ovlivňují zdravý životní styl, poskytne aplikace doporučení na alternativní zdravější varianty. Pro uživatele mohou být tato doporučení a tipy zdrojem podpory a motivace.

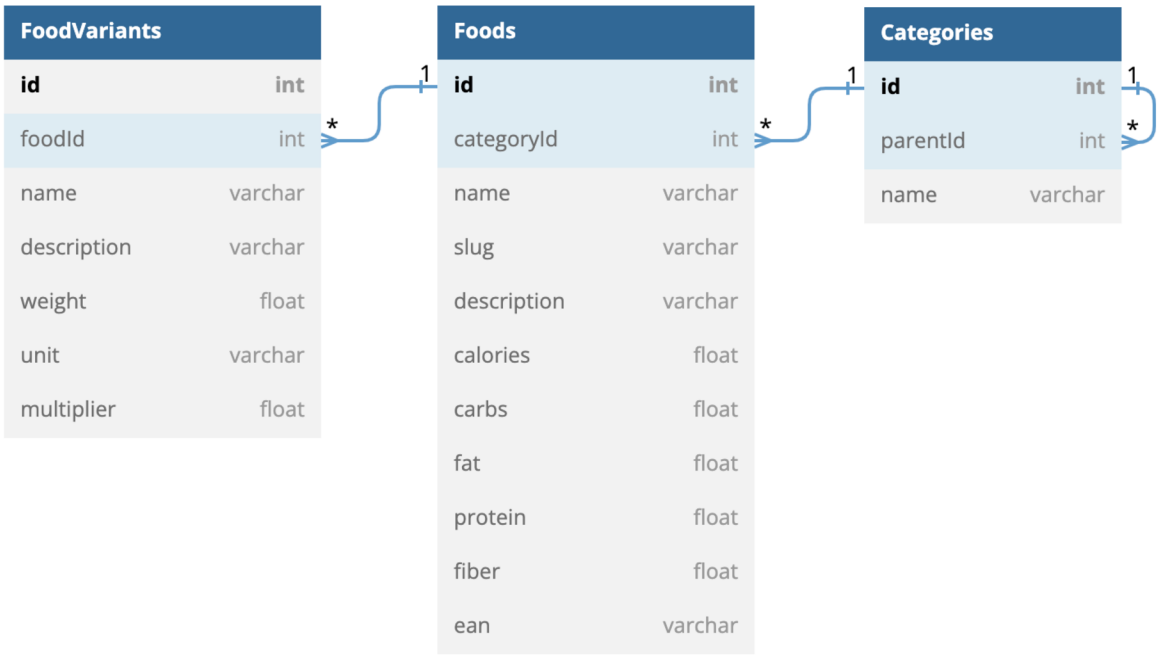
8 Implementace řešení

Následující kapitola popisuje implementaci řešení, které byly navrženy v předchozí kapitole s důrazem na ukázku praktických aspektů vývoje s použitím dříve uvedených softwarových nástrojů, technologií a programovacích jazyků.

8.1 Databáze potravin

Databáze potravin představuje jednu z klíčových částí aplikace. Jejím účelem je poskytnout uživatelům přístup k detailním informacím o potravinách, které konzumují, jako například množství kalorií a hodnoty makroživin (sacharidy, bílkoviny, tuky). Díky tomu mohou uživatelé přesněji plánovat svůj stravovací režim a sledovat pokrok v rámci splnění stanovených cílů. Vyhledávat potraviny v databázi je možné pomocí kategorií a klíčových slov, které reprezentují název potraviny. Výsledkem vyhledávání je seznam potravin, které odpovídají zadaným parametrům. Každému uživateli jsou dále k dispozici individuální seznamy oblíbených a nejčastěji zapisovaných potravin.

Potraviny jsou organizovány v tabulce *Foods* (Obrázek 10). Záznam představuje konkrétní potravinu a její množství kalorií a hodnoty makroživin na 100 gramů. Atributy jsou dále popsány v následující tabulce (Tabulka 3).



Obrázek 10 –Schéma databáze potravin (vlastní zpracování)

Tabulka 3 - Popis tabulky Foods

Tabulka *Foods*	
Atribut	**Význam**
id	Unikátní identifikátor potraviny
categoryId	Identifikátor kategorie, do které potravina náleží
name	Název potraviny
slug	Název potraviny bez diakritiky a mezer
description	Detailní popis potraviny
calories	Celkové množství kalorií na 100 g
carbs	Celkové množství sacharidů na 100 g
fat	Celkové množství tuků na 100 g
protein	Celkové množství bílkovin na 100 g
fiber	Celkové množství vlákniny na 100 g
ean	Textová reprezentace čárového kódu potraviny

V rámci zpříjemnění používání databáze potravin je vhodné u každé potraviny evidovat různé varianty porcí. K tomuto účelu slouží tabulka *FoodVariants*. Každá potravina v tabulce *Foods* obsahuje minimálně dvě varianty v tabulce *FoodVariants*, a to 1 gram a 100 gramů. Další varianty jsou přidány individuálně dle konkrétní potraviny. Například v případě toustového chleba existují varianty *kus (25 g)* nebo *balení (500 g)*, balená voda může mít varianty *250 ml*, *500 ml*, *1 l* nebo *1,5 l*. Pro získání přesného množství kalorií a makroživin je použita číselná konstanta *multiplier*. Součinem konstanty a původních hodnot jsou získány nové hodnoty pro konkrétní variantu. Výpočet kalorií je blíže znázorněn v následující tabulce (Tabulka 4). Výpočet bílkovin, sacharidů a tuků probíhá analogicky stejným způsobem.

Tabulka 4 - Ukázka použití konstanty Multiplier

Váha potraviny	**Konstanta**	**Množství kalorií**
100 g	100 / 100 = 1	1 × 230 = 230 kcal
250 g	250 / 100 = 2,5	2,5 × 230 = 575 kcal
777 g	777 / 100 = 7,77	7,77 × 230 = 1 787 kcal

Atributy tabulky *FoodVariants* jsou popsány v následující tabulce (Tabulka 5).

Tabulka 5 - Popis tabulky FoodVariants

Tabulka *FoodVariants*	
Atribut	**Význam**
id	Unikátní identifikátor varianty
foodId	Identifikátor potraviny, ke které varianta náleží
name	Název varianty
description	Detailní popis varianty
weight	Hmotnost varianty
unit	Jednotka použitá k vyjádření hmotnosti (g, kg, ml, …)
multiplier	Číselná konstanta použitá pro vynásobení základních hodnot potraviny na 100 gramů, tím je získána cílená hodnota na požadovanou hmotnost

V rámci organizace jsou potraviny uspořádány do 11 kategorií, které se dále dělí na 52 podkategorií. Například hlavní kategorie *Nápoje* je rozdělena na podkategorie *Alkoholické*, *Nealkoholické* a *Instantní*, a kategorie *Ovoce* pak obsahuje podkategorie *Čerstvé*, *Sušené* a *Mražené*. Každá potravina je zařazena do jedné konkrétní podkategorie. Atributy jsou popsány v následující tabulce (Tabulka 6).

Tabulka 6 - Popis tabulky Categories

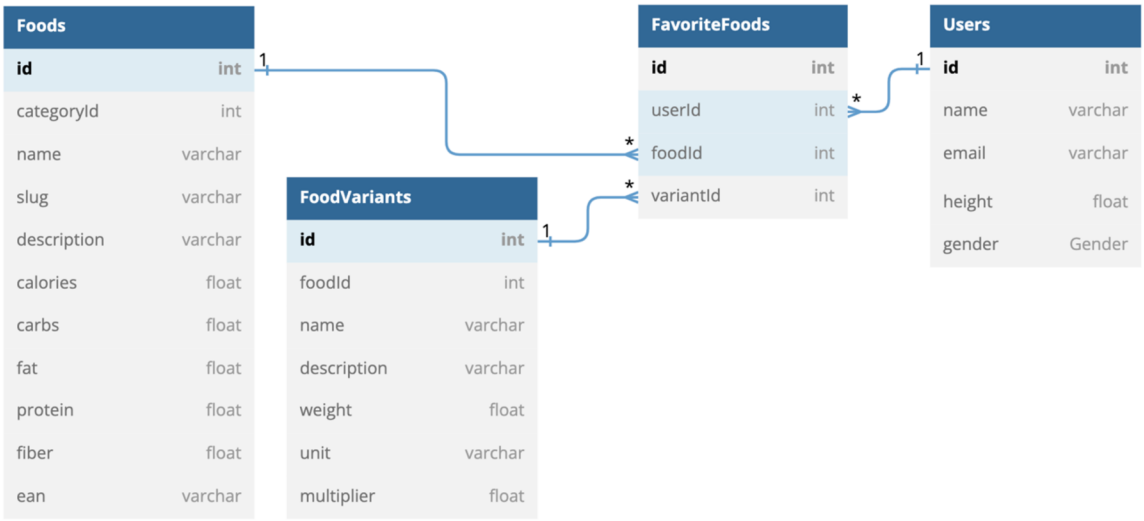
Tabulka *Categories*	
Atribut	**Význam**
id	Unikátní identifikátor kategorie
parentId	Identifikátor nadřazené kategorie
name	Název kategorie

Aby uživatelé mohli jednoduše a rychle zapisovat potraviny do svého jídelníčku, mohou používat funkci oblíbených potravin. Funkce umožňuje uživatelům snadno označit potraviny a nápoje jako oblíbené. Ty se následně v aplikaci zobrazí ve speciální sekci, která je snadno dostupná a umožňuje uživatelům rychlý přístup k potravinám, které často konzumují. Následně mohou tyto potraviny vybrat a přidat je do svého jídelníčku. Všechny oblíbené potraviny uživatelů jsou organizovány v tabulce *FavoriteFoods* (Obrázek 11). Atributy jsou popsány v následující tabulce (Tabulka 7).

Tabulka 7 - Popis tabulky FavoriteFoods

Tabulka *FavoriteFoods*	
Atribut	**Význam**
id	Unikátní identifikátor oblíbené potraviny
userId	Identifikátor uživatele
foodId	Identifikátor potraviny
variantId	Identifikátor varianty

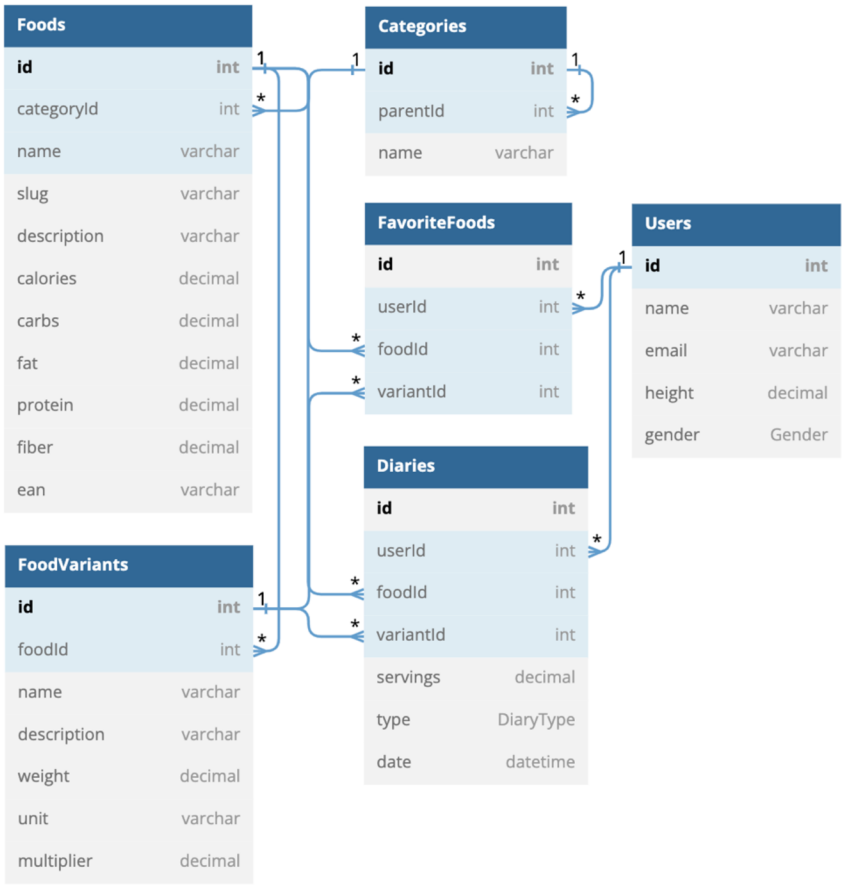
Díky funkci oblíbených potravin se uživatelé nemusí vždy spoléhat na vyhledávání potravin v databázi, což značně usnadňuje a zrychluje proces zapisování. Funkce má také výhodu v tom, že uživatelé mohou snadno sledovat své oblíbené potraviny a analyzovat, jaký vliv mají na jejich kalorický příjem.



Obrázek 11 – Schéma databáze potravin a tabulka oblíbené (vlastní zpracování)

Současně s funkcí oblíbených potravin má uživatel k dispozici také seznam těch nejčastěji zapisovaných. Uživatelům poskytuje užitečné informace o stravovacích návycích. Mohou sledovat, jaké potraviny konzumují nejčastěji a zjistit, zda jsou jejich stravovací návyky zdravé a vyvážené. Pokud zjistí, že konzumují příliš mnoho kalorií z určitých potravin, mohou snadno najít zdravější alternativy a náležitě upravit svůj jídelníček. Obě zmíněné funkce poskytují uživatelům možnost navrhnout si personalizovaný jídelníček, který jim vyhovuje, a díky tomu si vytvořit zdravější stravovací návyky.

Kompletní schéma databáze potravin (Obrázek 12) obsahuje celkem šest tabulek s příslušnými vzájemnými relacemi, aby bylo dosaženo požadované funkcionality.



Obrázek 12 - Kompletní schéma databáze potravin (vlastní zpracování)

Záznamy o potravinách jsou organizovány v relačním databázovém systému PostgreSQL, který poskytuje techniku fulltextového vyhledávání (fulltext search). Na rozdíl od pouhého porovnávání slov podle určitého vzoru nebo masky je fulltextové vyhledávání přesnější a komplexnější. Umožňuje hledat slova nebo fráze v kompletním textovém dokumentu a je navrženo tak, aby umělo pracovat s výrazovými vztahy mezi slovy, například synonyma nebo kořeny slov. Fulltextové vyhledávání je jazykově závislé a spoléhá na určitou úroveň porozumění jazyka pro získání sémanticky smysluplných výsledků.

Vyhledávání v tabulce *Foods* je prováděno nad atributem *name*, který definuje název potraviny ve formě datového typu *text* (Tabulka 3). Ten je nutné převést na datový typ *tsvector*, který je vhodný pro efektivní fulltextové vyhledávání. Funkce *to_tsvector* přijímá dva parametry. První definuje použitý slovník a druhý parametr

je samotný text. Výsledkem je seznam normalizovaných slov, který se označuje jako *document* [46]. Na příkladu (Obrázek 13) lze vidět, že slova v množném čísle byla převedena na tvar jednotný a zároveň byla odstraněna takzvaná stop-slova jako jsou spojky, předložky nebo zájmena, která neovlivňují výsledek vyhledávání, protože neposkytují specifické informace o obsahu textu. Čísla u jednotlivých slov představují pozici v původním textu.

```
SELECT * FROM to_tsvector('cs', 'Lívance s banány a kokosem');

-- 'banán':3 'kokos':5 'lívanec':1
```

Obrázek 13 - Ukázka použití funkce to_tsvector (vlastní zpracování)

Následně je nutné převést hledaný text na fulltextový dotaz datového typu *ts_query*. K tomu slouží funkce *plainto_tsquery*, která přijímá dva parametry. První definuje použitý slovník a druhým parametrem je samotný text. Výsledkem je seznam normalizovaných klíčových slov v kombinaci s operátory &, kterými jsou slova spojena (Obrázek 14). Seznam se označuje jako *query* [46].

```
SELECT * FROM plainto_tsquery('cs', 'lívanec s banánem');

-- 'lívanec' & 'banán'
```

Obrázek 14 - Ukázka použití funkce plainto_tsquery (vlastní zpracování)

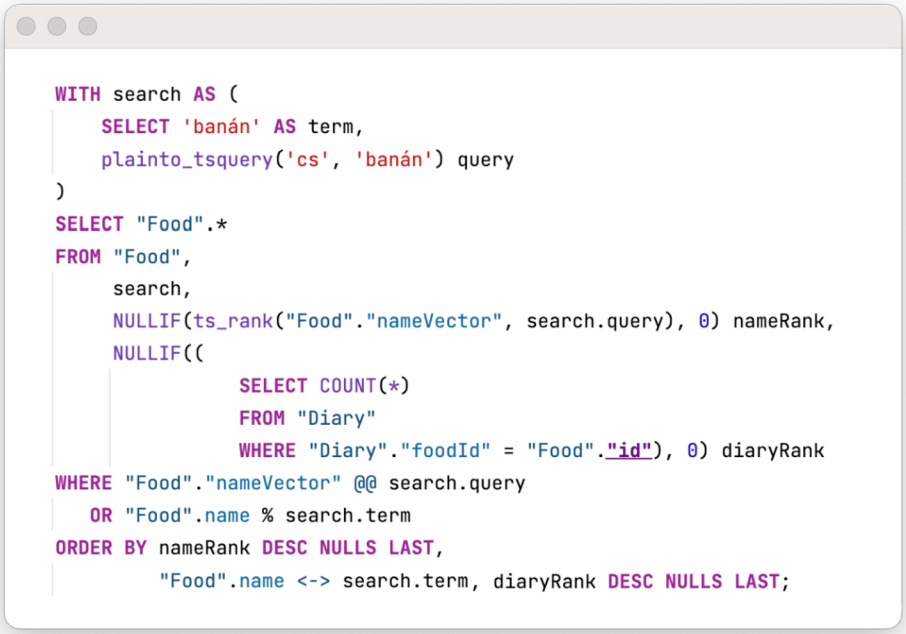
Po získání seznamů *document* a *query* je možné provést samotné fulltextové vyhledávání pomocí operátoru @@, který porovná *query* a *document* a vrátí výsledek ve formě pravdivostní hodnoty *true* (ano) nebo *false* (ne) [47]. V případě

uvedeného příkladu je výsledkem hodnota true, protože v původním textu došlo k nalezení shody ve slovech *lívanec* a *banán*.

V případě většího množství nalezených shod je možné výsledky seřadit podle skóre, které určuje, jak je nalezená shoda relevantní. K tomu je možné použít funkci *ts_rank*, která přijímá parametry *document* a *query*. Výsledkem je hodnota v intervalu <0; 1>, která udává míru relevantnosti [48].

U mobilních aplikací se uživatelé při psaní na klávesnici běžně setkávají s nechtěnými překlepy. Pokud by se slova v předchozím příkladu lišila, byť jen v jediném znaku, nemuselo by dojít ke správné normalizaci slov a tím by byly výsledky vyhledávání nepřesné. PostgreSQL databáze poskytuje operátor *%* k porovnávání dvou textových řetězců a vrací pravdivostní hodnotu (true nebo false), která určuje, zda jsou si řetězce podobné [49]. Dále je možné stejným způsobem využít operátor <->. Výsledkem je číslo v intervalu <0;1>, které udává míru podobnosti slov. Proces vyhledávání lze rozšířit o tuto hodnotu. V případě, kdy nedojde k nalezení shody pomocí porovnání *query* a *document*, je možné využít hodnotu podobnosti řetězců.

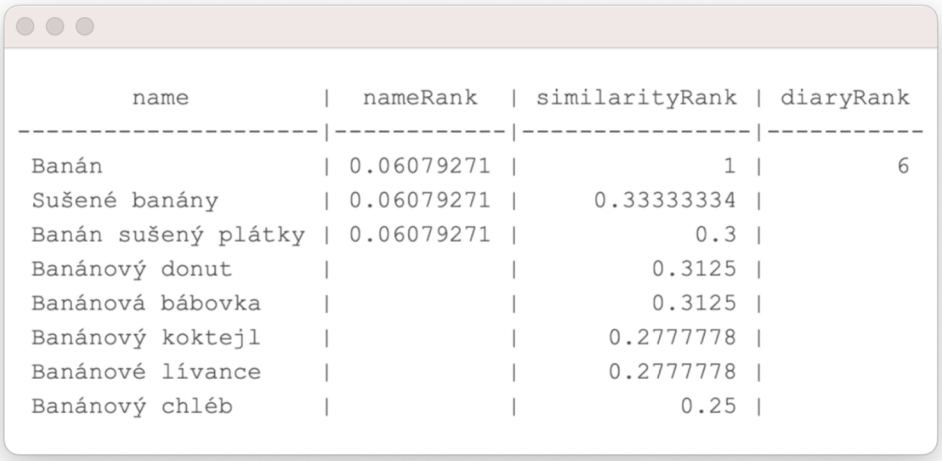
Rychlost provedení sestaveného dotazu je průměrně 700 ms. To je pro běžné používá mobilní aplikace poměrně vysoká hodnota. Výkon dotazu lze optimalizovat přidáním atributu *nameVector* do tabulky *Foods*. Hodnotou bude výsledek funkce *to_tsvector*. Díky tomu je možné v dotazu využít předem připravenou hodnotu a není nutné funkci pro každý záznam opakovaně volat. Dalším krokem optimalizace je přidání indexů atributům *name* a *nameVector*. Rychlost dotazu se po optimalizaci výrazně zlepšila a dosahuje hodnoty 6 ms.



```
WITH search AS (
    SELECT 'banán' AS term,
    plainto_tsquery('cs', 'banán') query
)
SELECT "Food".*
FROM "Food",
     search,
     NULLIF(ts_rank("Food"."nameVector", search.query), 0) nameRank,
     NULLIF((
                SELECT COUNT(*)
                FROM "Diary"
                WHERE "Diary"."foodId" = "Food"."id"), 0) diaryRank
WHERE "Food"."nameVector" @@ search.query
   OR "Food".name % search.term
ORDER BY nameRank DESC NULLS LAST,
         "Food".name <-> search.term, diaryRank DESC NULLS LAST;
```

Obrázek 15 - Ukázka fulltextového vyhledávání (vlastní zpracování)

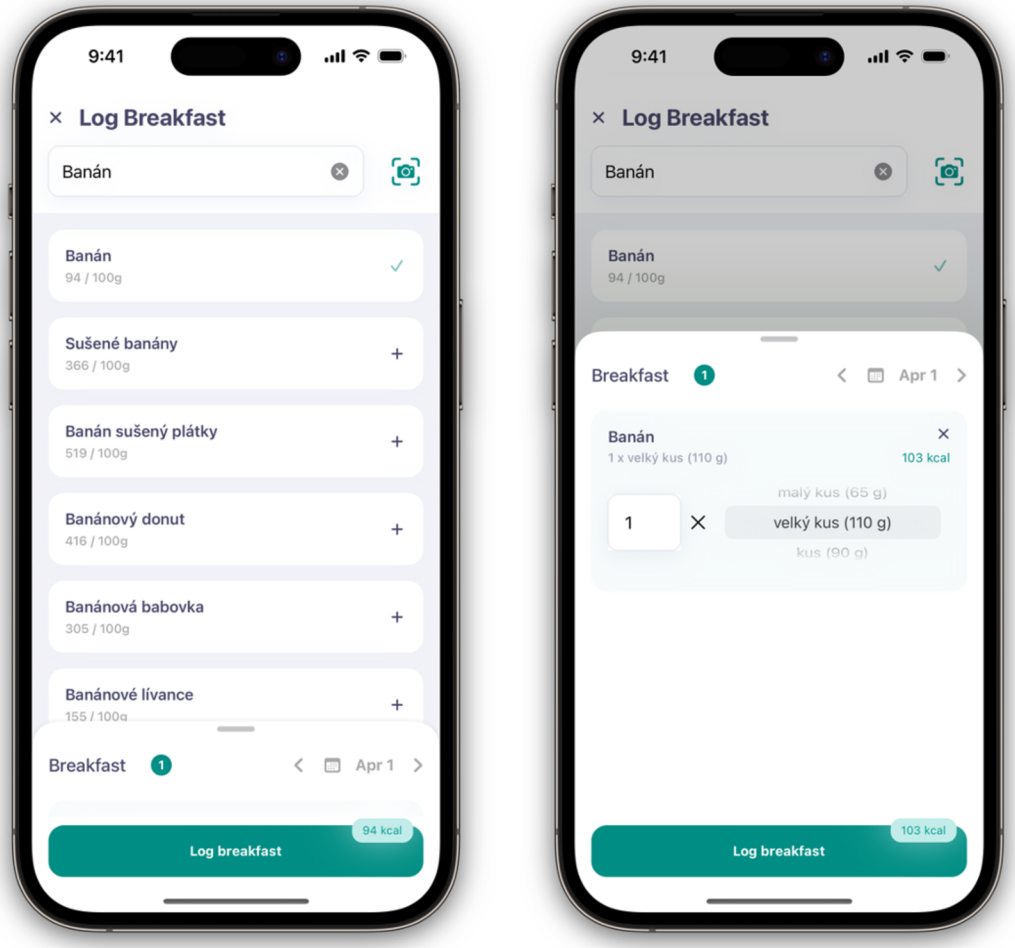
Je-li provedeno vyhledávání potraviny pro klíčové slovo *banán*, je nalezeno osm relevantních záznamů, které jsou seřazeny sestupně podle hodnot skóre *nameRank* a *similarityRank*. Na příkladu (Obrázek 16) lze vidět, že k přesné shodě dle klíčového slova došlo v prvních třech případech. Ostatní záznamy byly nalezeny díky doplňkovému skóre podobnosti slov *banán* a *banánový*. Ke každému záznamu je počítána hodnota *diaryRank*, která říká, kolikrát byla daná potravina zapsána všemi uživateli do jídelníčku. Lze předpokládat, že uživatel bude vyhledávat potraviny, které jsou ostatními uživateli také vyhledávány a díky hodnotě skóre *diaryRank* může být potravina v seznamu zvýhodněna a zobrazena výše než ostatní.



name	nameRank	similarityRank	diaryRank
Banán	0.06079271	1	6
Sušené banány	0.06079271	0.33333334	
Banán sušený plátky	0.06079271	0.3	
Banánový donut		0.3125	
Banánová bábovka		0.3125	
Banánový koktejl		0.2777778	
Banánové lívance		0.2777778	
Banánový chléb		0.25	

Obrázek 16 - Výsledek fulltextového vyhledávání (vlastní zpracování)

V aplikaci má uživatel k dispozici vyhledávácí vstupní pole, ve kterém specifikuje název hledané potraviny. Během zadávání znaků aplikace odesílá požadavek na server, který provede fulltextové vyhledávání v databázi a aplikaci vrátí seznam nalezených potravin včetně variant (Obrázek 17). Uživatel může ze seznamu vybrat potravinu, následně konkrétní variantu a hodnoty uložit do jídelníčku. Proces vyhledávání zároveň implementuje podporu stránkování. Seznam potravin, který server vrací, je omezen na 30 položek. Pokud uživatel prochází seznamem směrem dolů a přibližuje se jeho konci, jsou automaticky načítány další stránky po 30 položkách a přidávány k aktuálnímu obsahu seznamu. Jedná se o techniku zvanou *Infinity scroll*, která umožňuje uživatelům kontinuálně procházet obsah bez potřeby kliknutí na tlačítko *Další stránka* nebo *Načíst více*.



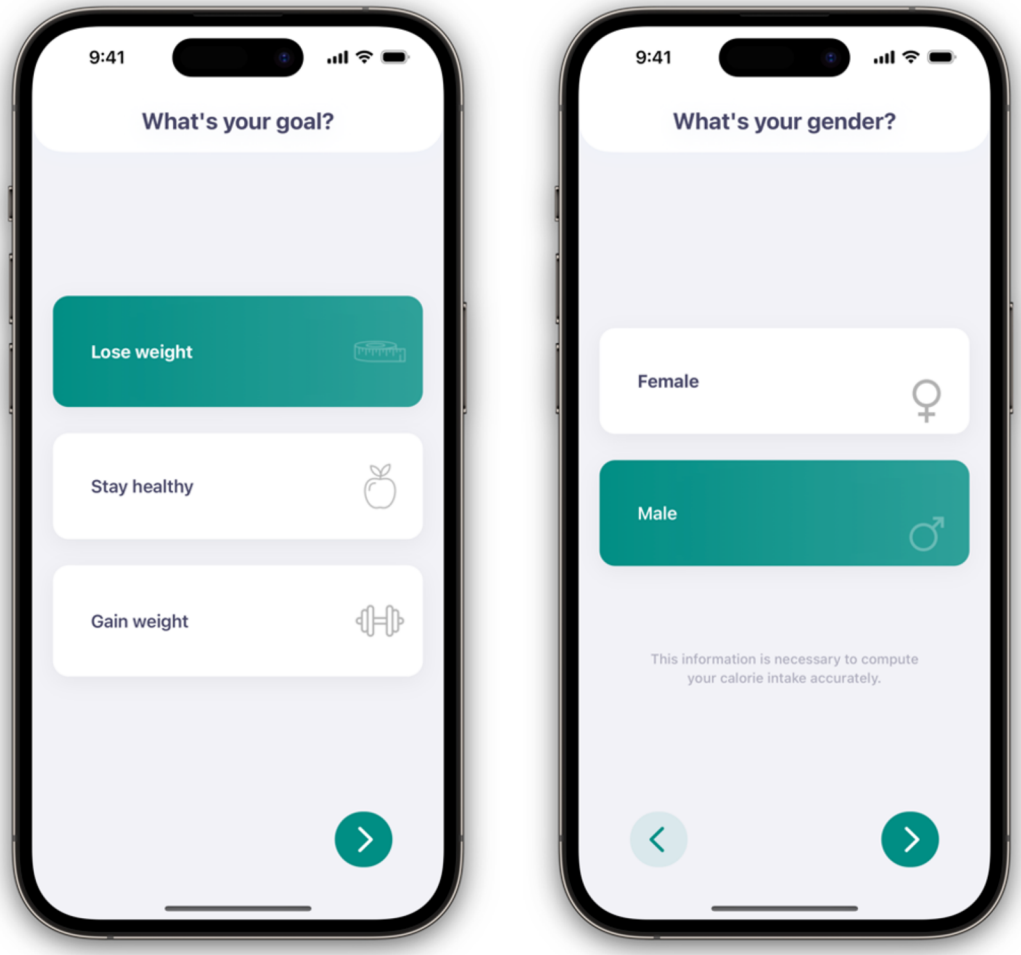
Obrázek 17 - Vyhledávání potravin a seznam výsledku (vlevo),
zápis potraviny a její varianty do jídelníčku (vpravo) (vlastní zpracování)

8.2 Sledování kalorického příjmu

Než je možné aplikaci používat, musí uživatel vyplnit vstupní dotazník, který obsahuje sérii jednoduchých otázek týkajících se zdraví, životního stylu, fyzické aktivity a výživových preferencí (Tabulka 8). Získaná data se použijí pro vytvoření kalorického a výživového plánu dle individuálních potřeb uživatele.

Tabulka 8 - Údaje vstupního dotazníku

Údaj	**Přípustné hodnoty**
Cíl	Zhubnout Být fit Nabrat svaly
Pohlaví	Žena Muž
Věk	13–99 let
Výška	120–220 cm
Váha	30–220 kg
Úroveň aktivity	Sedavý způsob života Mírně aktivní Středně aktivní Velmi aktivní Vysoce aktivní



Obrázek 18 - Vstupní dotazník, výběr cíle (vlevo), výběr pohlaví (vpravo) (vlastní zpracování)

Obrázek 19 - Vstupní dotazník, volba věku (vlevo),
výběr tělesné výšky (vpravo) (vlastní zpracování)

Dle zadané výšky a pohlaví probíhá výpočet rozmezí doporučené váhy s použitím rovnic Hamwi a Miller (Obrázek 18). Využívají se pro odhad ideální tělesné hmotnosti pro ženy a muže v závislosti na výšce (Obrázek 19). Jedná se však o hrubý odhad, protože nejsou zohledněny faktory jako svalová hmota, tělesný tuk, věk nebo fyzická aktivita. Rovnice byly navrženy pro lékaře, aby jim pomohly při rychlém posuzování tělesné hmotnosti pacientů [50].

$$Hamwi\ ženy = 45.5 + (2.2 \times počet\ palců\ přes\ 5\ stop\ výšky) \quad (7)$$

$$Hamwi\ muži = 48 + (2.7 \times počet\ palců\ přes\ 5\ stop\ výšky) \quad (8)$$

$$Miller\ ženy = 53.1 + (1.36 \times počet\ palců\ přes\ 5\ stop\ výšky) \quad (9)$$

$$Miller\ muži = 56.2 + (1.41 \times počet\ palců\ přes\ 5\ stop\ výšky) \quad (10)$$

Výsledné hodnoty obou rovnic jsou zobrazeny uživateli jako doporučení formou spodní a horní hranice při výběru tělesné hmotnosti a mají informativní charakter (Obrázek 20). Dle výšky a váhy je spočtena hodnota BMI, která poskytuje přehled o tom, zda je váha uživatele v normálním rozmezí nebo hrozí možná zdravotní rizika spojená s podváhou nebo nadváhou.

Na základě údajů pohlaví, věk, výška a váha je spočtena hodnota bazálního metabolismu dle Mifflin-St Jeorovi rovnice. Jedná se o množství energie, kterou tělo potřebuje v klidovém stavu na udržení základních životních funkcí jako je dýchání, krevní oběh, trávení a udržování tělesné teploty.

Doporučený celkový energetický příjem je výsledkem součinu hodnoty bazálního metabolismu a konstanty dle úrovně fyzické aktivity, kterou uživatel zvolil dle svých pohybových preferencí (Tabulka 2) (Obrázek 20). Následně je hodnota upravena součinem s konstantou dle vybraného cíle a tím je získána konečná hodnota celkového energetického příjmu dle individuálních potřeb uživatele (Tabulka 9). Pokud je cílem hubnout, hodnota bude menší než doporučená, čímž se

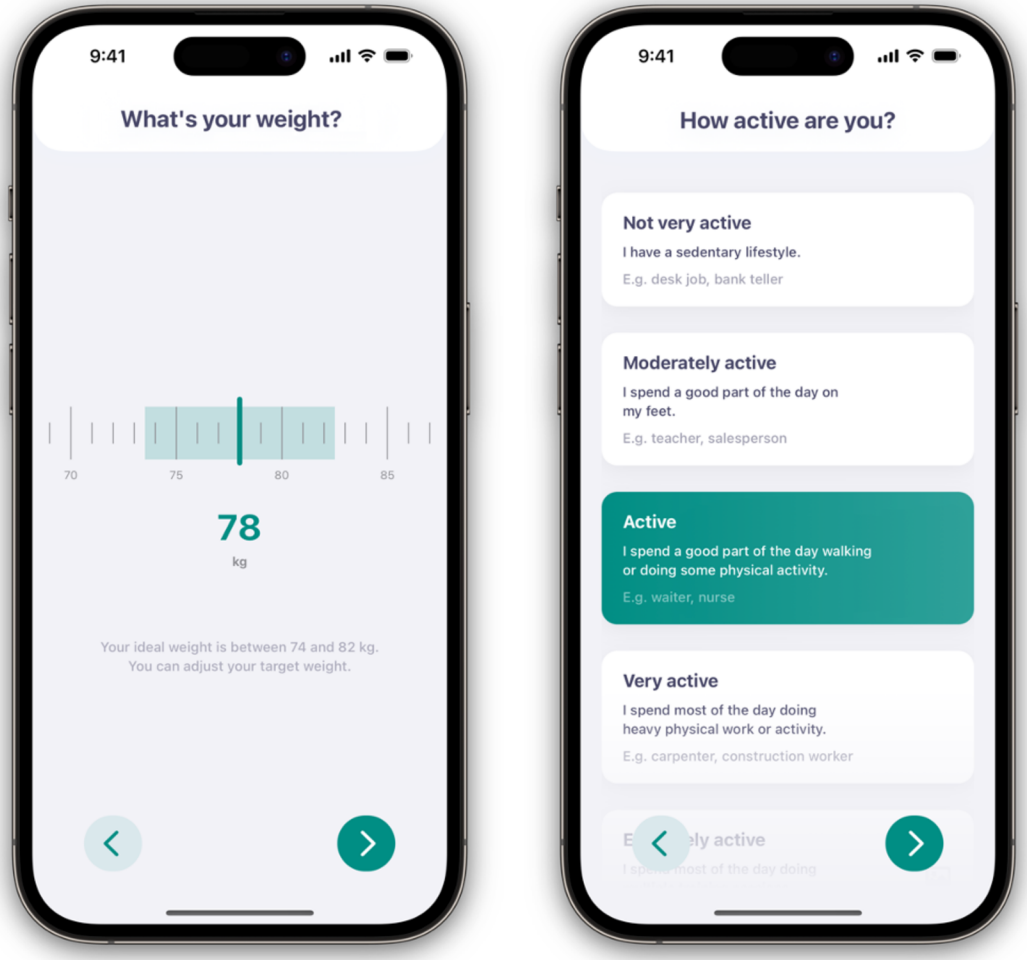
docílí mírného kalorického deficitu, který je pro hubnutí nezbytný. Jeli cílem nabrat svaly, konečná hodnota energetického příjmu bude vyšší než doporučená, aby se docílilo kalorického nadbytku, který je současně se silovým cvičením pro růst svalové hmoty nezbytný.

Tabulka 9 - Konstanty dle cíle

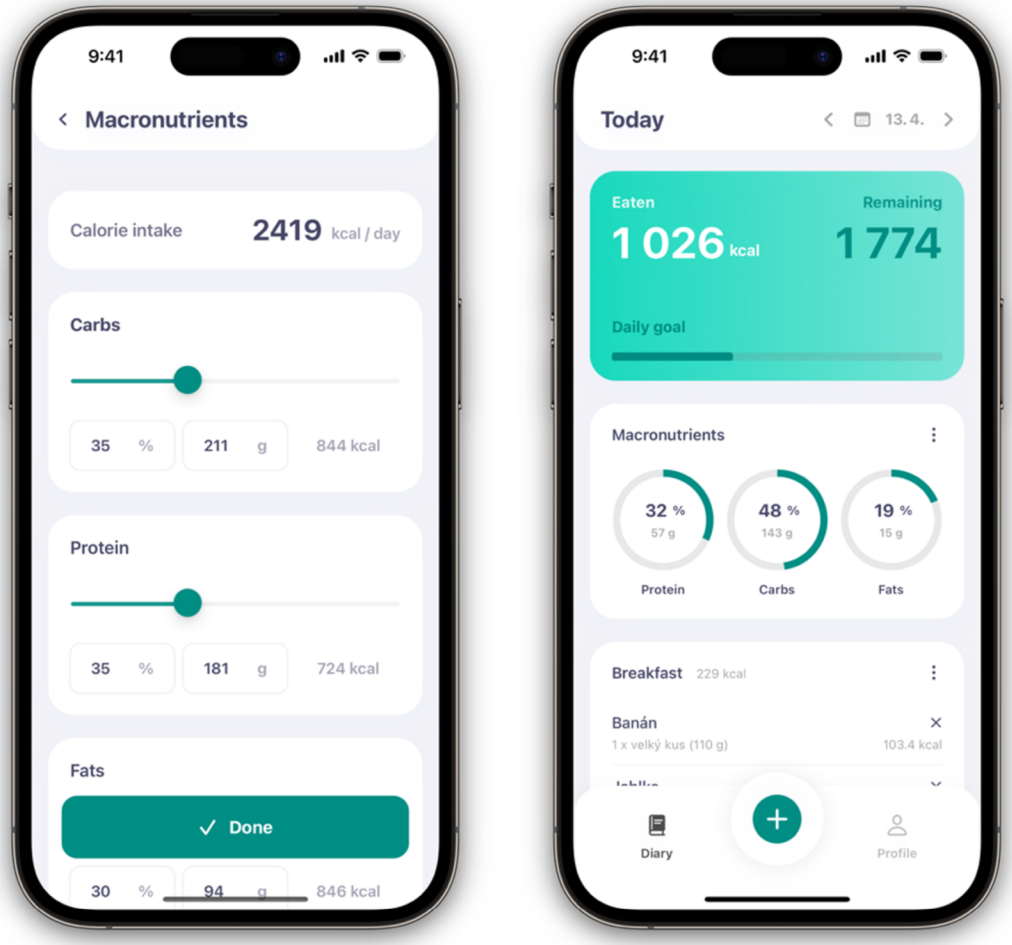
Konstanta	**Cíl**
0.85	Zhubnout
0.985	Být fit
1,1	Nabrat svaly

Následně aplikace vypočítá doporučený příjem makroživin. Denní příjem sacharidů by měl tvořit 45–65 %, tuků 20–35 % a bílkovin 10–35 % z hodnoty celkového kalorického příjmu. Ve výchozím nastavení vytvoří aplikace plán, který obsahuje 35 % sacharidů, 30 % tuků a 35 % bílkovin (Obrázek 21). Uživatelé mají možnost rozložení hodnot upravit dle svých preferencí, platí však omezení, že jednotlivé hodnoty nepřekročí doporučený denní příjem a součet energetické hodnoty makroživin není vyšší než celkový příjem kalorií.

Po dokončení vstupního dotazníku se hodnoty celkového energetického příjmu a makroživin pro konkrétního uživatele odešlou z aplikace na server, kde proběhne jejich uložení do databáze. Pokud uživatel zapíše potravinu do svého jídelníčku, jsou na základě těchto potravin vypočítány aktuální hodnoty celkového energetického příjmu a makroživin. Uživatel má tak přehled o tom, jakých pokroků dosahuje během dodržování jídelníčku. K dispozici jsou údaje o počtu kalorií, které zbývají a procentuální hodnoty makroživin, které má uživatel dle svého individuálního plánu dodržet (Obrázek 21).



Obrázek 20 - Vstupní dotazník, volba tělesné váhy (vlevo), výběr úrovně fyzické aktivity (vpravo) (vlastní zpracování)



Obrázek 21 - Plán příjmu kalorií a makroživin dle vstupního dotazníku (vlevo), aktuální hodnoty dle zapsaných potravin (vpravo) (vlastní zpracování)

8.3 Skenování čárových kódů

Funkce skenování čárových kódů usnadňuje a zrychluje zapisování potravin do jídelníčku, protože uživatelé nemusí manuálně zadávat informace o potravině, kterou vyhledávají. Když uživatel skenuje čárový kód na balení potraviny, aplikace automaticky vyhledá danou potravinu v databázi, a v případě shody uživateli potravinu zobrazí s možností zápisu do jídelníčku. Díky tomu zároveň dochází ke snížení chybovosti při zadávání potravin do aplikace. Implementace této funkce zahrnuje využití frameworku AVFoundation a knihovny Vision.

AVFoundation je součástí platformy iOS a poskytuje nástroje a rozhraní pro práci multimediálními daty. Skrze třídu *AVCaptureSession* je možné na zařízeních s operačním systémem iOS přistoupit ke vstupním zařízením jako jsou fotoaparáty, videokamery nebo mikrofony. Díky tomu je možné nahrávat video, pořizovat fotografie nebo zobrazovat živý náhled z kamery. K inicializaci třídy *AVCaptureSession* je nutné nastavit vstupní zařízení jako zdroj dat. To je možné pomocí třídy *AVCaptureDevice*, která reprezentuje konkrétní fyzické vstupní zařízení, jako například fotoaparát nebo mikrofon. Instance se následně předá do třídy *AVCaptureDeviceInput*, která poskytuje další možnosti konfigurace vstupního zařízení, například velikost rozlišení, snímkovací frekvence nebo úroveň světla a barevné tóny. Instance třídy *AVCaptureDeviceInput* je následně použita pro třídu *AVCaptureSession*. Tím je inicializace dokončena a je možné začít snímat data ze vstupního zařízení (Obrázek 22).

Celý proces čtení dat z nakonfigurovaného vstupního zařízení lze spustit pomocí funkce *startCapturing* (Obrázek 23), která jako parametr přijímá funkci *handler*. Ta se volá vždy, jsou-li k dispozici nová data ze vstupního zařízení. Výstupem funkce *handler* je snímek z kamery reprezentovaný jako objekt *UIImage*, který je možné zobrazit uživateli a následně s ním dále pracovat.

```swift
class CameraService {
    private var session: AVCaptureSession?
    private var output: AVCaptureVideoDataOutput?

    init() {
        let deviceSession = AVCaptureDevice.DiscoverySession(
            deviceTypes: [.builtInDualCamera],
            mediaType: .video,
            position: .back
        )

        if let input = try? AVCaptureDeviceInput(device: deviceSession.devices.first!) {
            let session = AVCaptureSession()
            session.sessionPreset = .high
            session.addInput(input)

            let output = AVCaptureVideoDataOutput()

            if session.canAddOutput(output) {
                session.addOutput(output)
                self.output = output
            }

            session.commitConfiguration()

            self.session = session
        }
    }
}
```

Obrázek 22 - Inicializace objektu AVCaptureSession

```swift
class CameraService {
    private var session: AVCaptureSession?
    private var output: AVCaptureVideoDataOutput?
    private var videoHandler: ((UIImage) -> Void)?

    init() { ••• }

    func startCapturing(_ handler: @escaping (UIImage) -> Void) {
        if let session = session {
            session.startRunning()
        }
        self.videoHandler = handler
    }

    func stopCapturing() {
        session?.stopRunning()
    }
}
```

Obrázek 23 - Implementace funkce startCapturing

Proces následně pokračuje identifikací čárového kódu na získaném snímku z kamery. K tomu je využita knihovna Vision, která je součástí platformy iOS a nabízí pokročilé metody zpracování obrazu, například rozpoznání obličeje nebo textu, detekce objektů a také skenování čárových a QR kódů. Snímek získaný z kamery je předán do třídy *VNImageRequestHandler*. Následně je možné zavolat funkci *perform* s parametrem *VNDetectBarcodeRequest*, který specifikuje požadovanou operaci se snímkem. V tomto případě bude funkce na snímku hledat čárový kód a v případě shody je možné přistoupit k textové podobě čárového kódu pomocí atributu *payloadStringValue* (Obrázek 24)

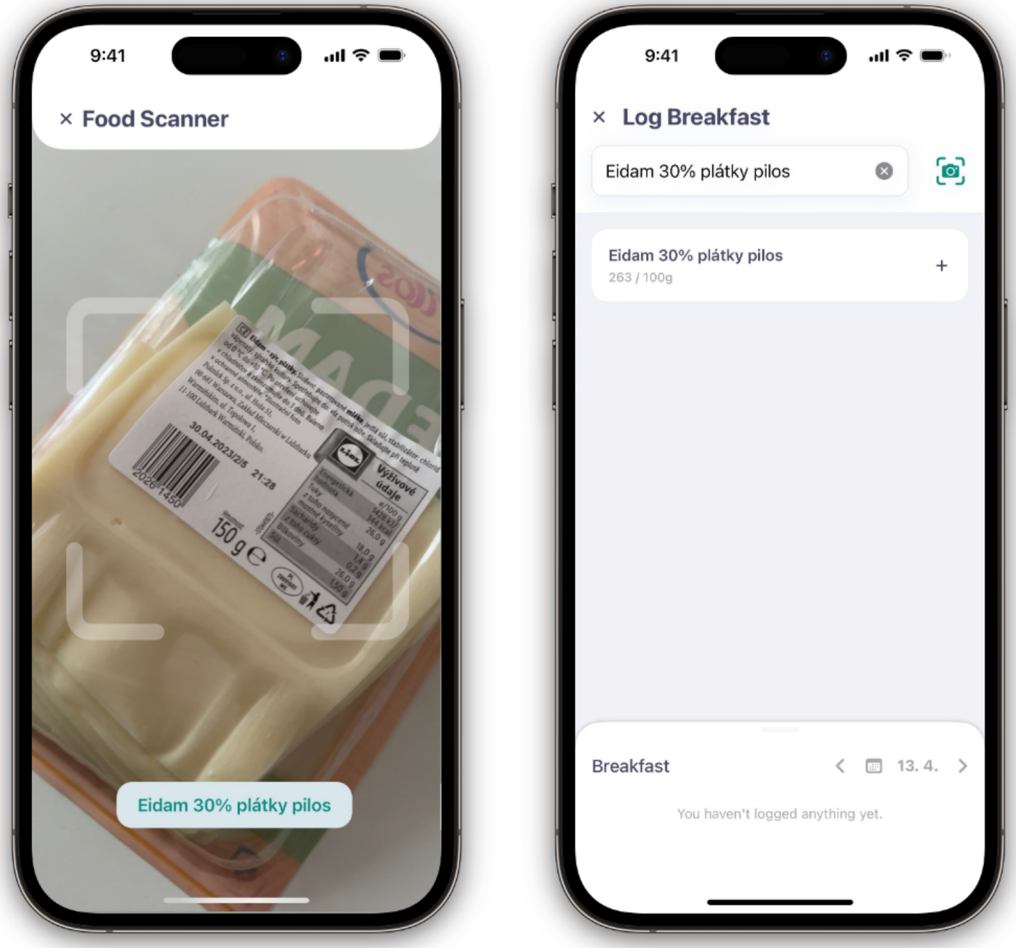
```
class BarcodeDetectionService: ObservableObject {

    func detectBarcode(image: CGImage, completetion: @escaping (String?) -> Void) {

        let handler = VNImageRequestHandler(cgImage: image)

        let handlerRequest = VNDetectBarcodesRequest(completionHandler: { request, _ in
            guard let result = request.results?.first else { return }

            if let result = result as? VNBarcodeObservation,
               let payload = result.payloadStringValue {
                completetion(payload)
            }
        })

        do {
            try handler.perform([handlerRequest])
        } catch {
            completetion(nil)
        }
    }
}
```

Obrázek 24 - Implementace detekce čárového kódu

Knihovna Vision je součástí frameworku CoreML pro strojové učení, který se nachází platformách iOS a macOS společnosti Apple a veškeré operace jsou prováděny modely umělých neuronových sítí [51].

Po získání textové hodnoty čárového kódu odešle aplikace požadavek na server, kde je provedeno vyhledávání v databázi potravin v tabulce *Foods* dle parametru *ean* (Obrázek 12). Pokud je potravina úspěšně nalezena, server vrací aplikaci odpověď ve formě záznamu potraviny včetně variant. V aplikaci je název potraviny zobrazen

uživateli s možností volby požadované varianty a následného zapsání do jídelníčku (Obrázek 25).



Obrázek 25 - Ukázka skenování čárového kódu a zobrazení výsledku (vlevo), vyhledávání dle výsledku (vpravo) (vlastní zpracování)

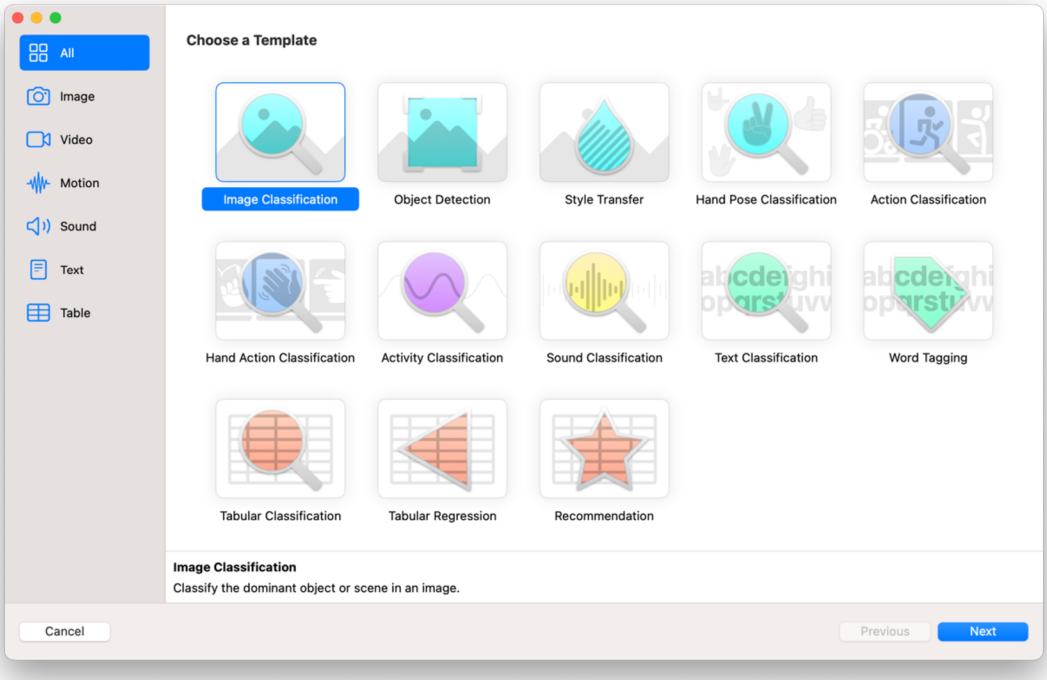
8.4 Rozpoznání potraviny z fotografie

Rozpoznání potraviny z fotografie představuje funkci, která uživatelům značným způsobem usnadní zapisování potravin do svého jídelníčku. Cílem je, aby uživatel pořídil fotografii potraviny a aplikace s využitím umělé neuronové sítě detekuje všechny možné potraviny, které by mohly odpovídat těm na pořízené fotografii, a seznam zobrazí uživateli.

Funkce je implementována prostřednictvím frameworku pro strojové učení CoreML, který je součástí platforem iOS a macOS společnosti Apple. V rámci ověření požadované funkcionality vznikne umělá neuronová síť, určená pouze k detekci ovoce a zeleniny. Připravený dataset zahrnuje 58 druhů ovoce a zeleniny a obsahuje celkem 2 900 fotografií. Objekty na fotografiích jsou zaznamenány v různých úhlech, pozicích, tvarech a úrovni osvětlení, ve snaze pokrýt co nejvíce možných variant z reálného prostředí. K vytvoření samotné neuronové sítě je použit nástroj CreateML, dostupný z frameworku CoreML. Jedná se o program určený k trénování vlastních modelů neurnových sítí v mnoha různých oblastech, například klasifikace fotografií, extrahování významu slov z textu nebo hledání vztahů a spojení mezi numerickými hodnotami. Pro účely detekce objektů na fotografiích je zvolena možnost klasifikace fotografií (Image Classification) (Obrázek 26).

Následně je nutné zvolit tři datasety – training, validation a testing. Jedná se o základní datové sady používané v procesu učení umělých neuronových sítí (Obrázek 27).

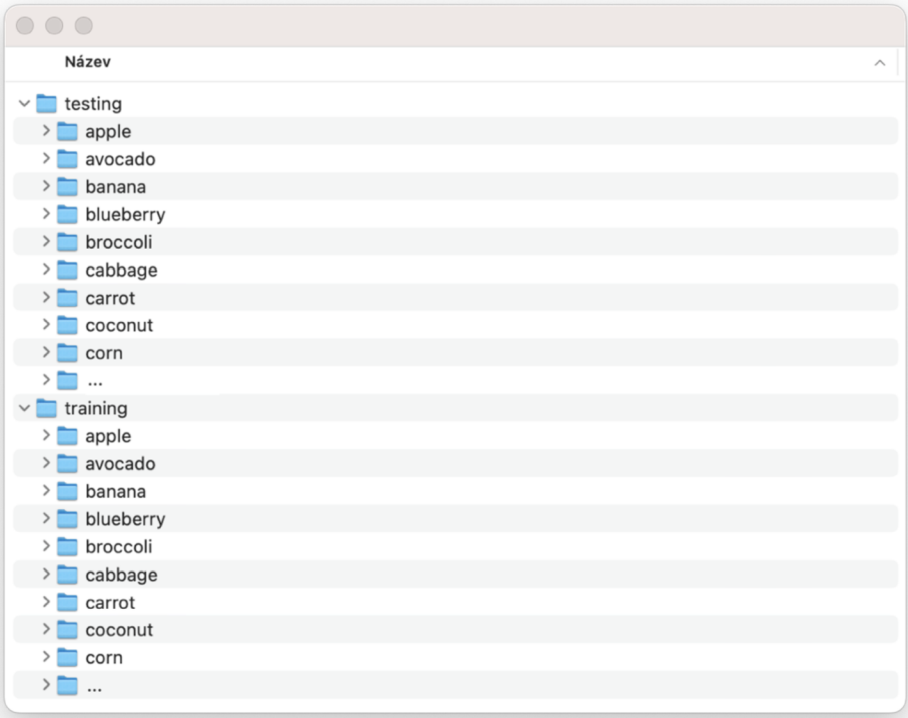
Dataset training (trénovací) obsahuje fotografie ovoce a zeleniny, které jsou organizovány ve vlastních adresářích podle názvu. Každý adresář pak obsahuje 30 fotografií konkrétní potraviny (Obrázek 28). Model umělé neuronové sítě tyto fotografie zpracovává, upravuje vnitřní hodnoty vah a prahů, a hledá rysy, podle kterých by bylo možné objekt na fotografii jednoznačně určit.



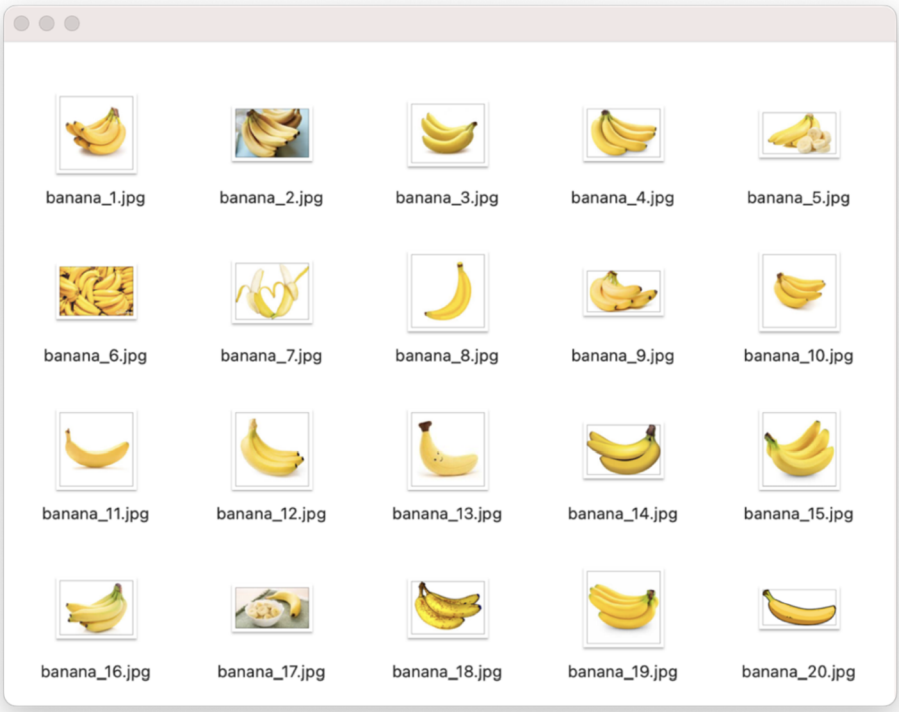
Obrázek 26 - Výběr zaměření neuronové sítě v CreateML (vlastní zpracování)

Dataset validation (ověřovací) obsahuje fotografie, které se používají pro ověření kvality modelu umělé neuronové sítě během procesu učení. Pokud model dosahuje až stoprocentní shody na trénovacích datech, může to být známka přeučení. To je možné odhalit právě na validačním datasetu, kde by takový model nedosahoval téměř žádné shody. Následně je nutné upravit parametry modelu s cílem zlepšit proces učení. Fotografie pro ověřovací dataset je možné dodat samostatně ve formě adresářů, stejně jako v případě trénovacího datasetu, nicméně nástroj CreateML umožňuje trénovací dataset automaticky rozdělit a 20 % fotografií použít pouze pro ověřovací dataset. Poslední varianta je také použita v případě tohoto modelu.

Training (testovací) dataset obsahuje fotografie organizované ve vlastních adresářích, stejně jako trénovací dataset. Jedná se o fotografie, které nebyly použity během trénování ani ověřování, tudíž se s nimi model zatím nesetkal. Díky tomu je možné ověřit, zda je model schopen sám a správně detekovat objekty na nových fotografiích bez toho, aniž by předem znal správný výsledek.



Obrázek 27 - Rozdělení použitého datasetu (vlastní zpracování)

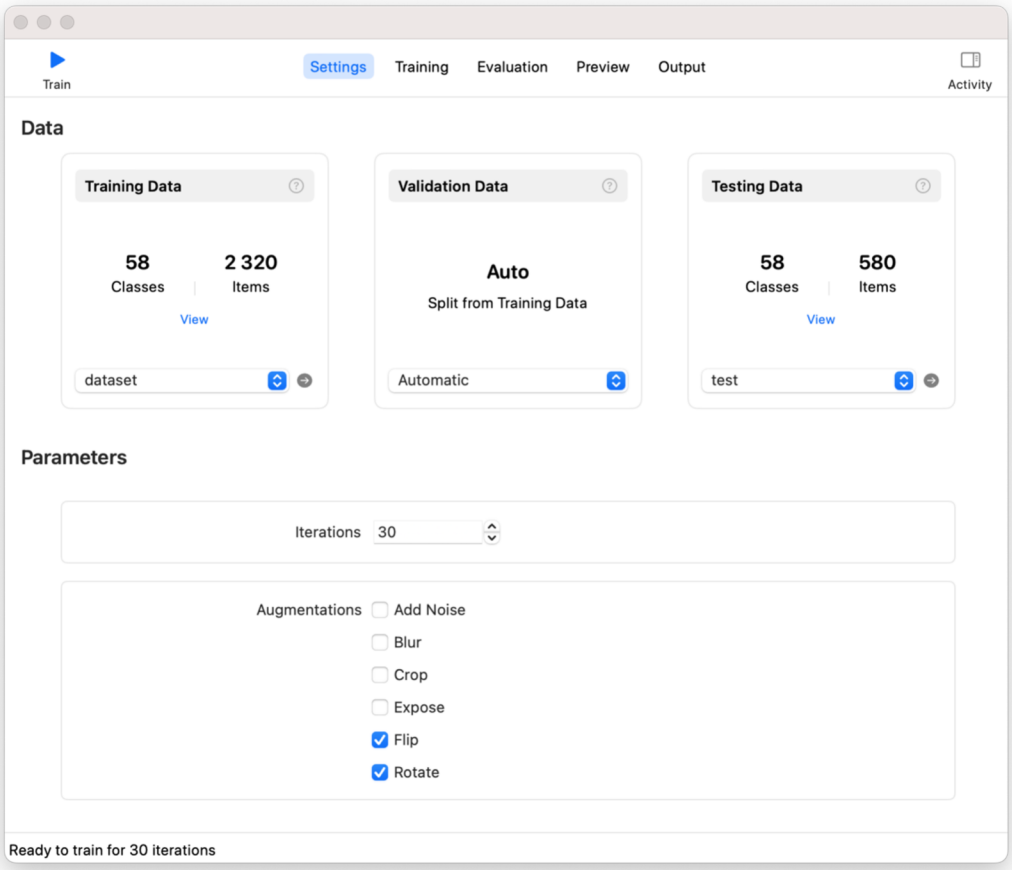


Obrázek 28 - Ukázka fotografií pro kategorii Banana (vlastní zpracování)

Nástroj CreateML nabízí řadu možností nastavení procesu učení umělé neuronové sítě (Obrázek 29). Parametr *iterations* (iterace) určuje kolikrát se bude opakovat proces trénování. CreateML využívá během učení algoritmus gradient descent, který v každé iteraci přizpůsobuje váhy modelu ve snaze minimalizovat chybu (rozdíl mezi předpovězenými a skutečnými výsledky). Počet opakování je důležitým parametrem modelu, protože může ovlivnit jeho výkon. Pokud je počet opakování příliš nízký, model se nemusí správně naučit rozpoznávat vzorce v datech a bude mít nízkou přesnost. Na druhé straně, pokud je počet opakování příliš vysoký, může se model přetrénovat, což znamená, že se naučí příliš dobře rozeznávat vzorce v trénovacích datech a nebude schopen správně pracovat s novými daty. Pro dosažení optimálního výkonu a úspěšnosti je nutné zvolit správnou hodnotu parametru *iterations*. V případě tohoto modelu se jako nejlepší hodnota jeví 30 opakování, neboť při nižších hodnotách dosahoval model velmi malé úspěšnosti detekce, a naopak při vyšších hodnotách dosahoval podobných výsledků úspěšnosti jako při hodnotě 30, ale za cenu potřeby delšího času a vyššího výpočetního výkonu během trénování, tudíž vyšší hodnoty nepřinesly výrazné zlepšení.

Další možností je umělá augmentace datasetu, neboli vytvoření kopií fotografií, které jsou určitým způsobem upraveny, aby se odlišovaly od originálů. CreateML dokáže fotografie náhodně otáčet, oříznout, upravit úrovně jasu, přidat šum nebo rozmazání. Pro účely použitého datasetu je zvolena augmentace pomocí náhodného otáčení fotografií.

Nástroj CreateML využívá techniku zvanou transfer learning (přenesené učení). Myšlenka spočívá v tom, že při vytváření modelu nezačínáme od úplného začátku, ale použijeme takzvaný před-trénovaný (pre-trained) model. V tomto případě se jedná o model VisionFeaturePrint navržený společností Apple a trénovaný na velkém množství fotografií různých kategorií [52].

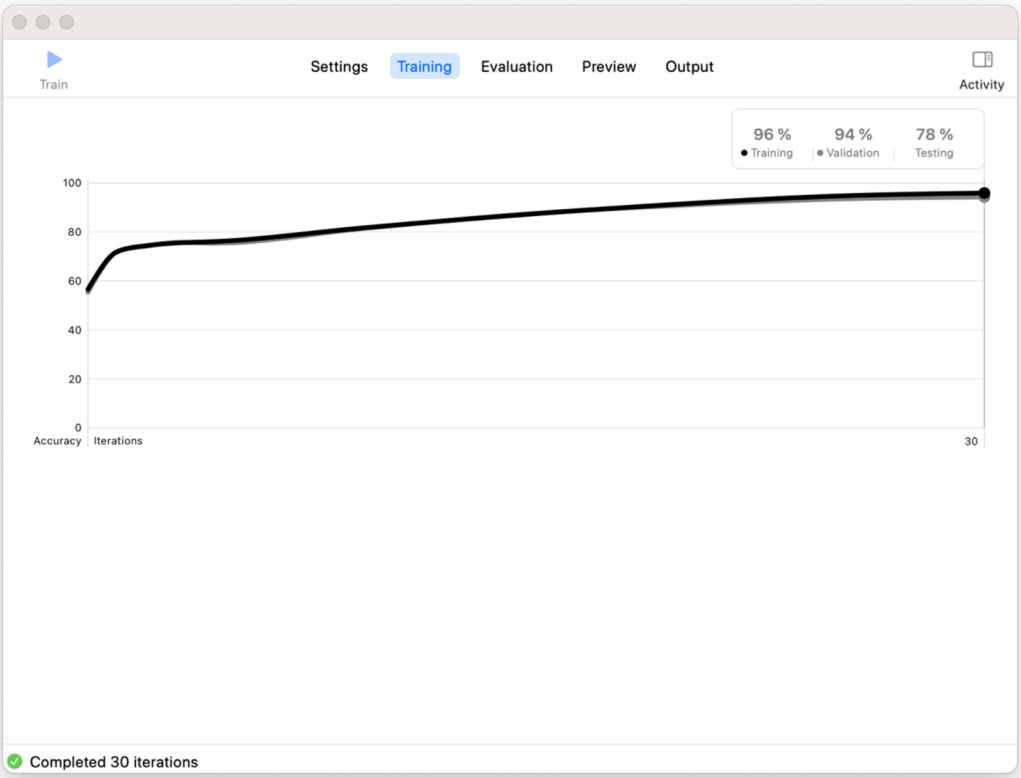


Obrázek 29 - Možnosti nastavení učení neuronové sítě (vlastní zpracování)

Přenesené učení funguje správně tehdy, jsou-li rysy vlastního datasetu fotografií dostatečně podobné těm rysům fotografií, které byly použity k prvotnímu trénování modelu neuronové sítě. Model je již naučen, jaké rysy má ve fotografii hledat a jak tyto rysy kombinovat, aby fotografii správně klasifikoval. Během trénování na datasetu fotografií ovoce a zeleniny už model ví, jaké rysy má hledat a extrahovat. Díky tomu je ušetřen výpočetní výkon a čas strávený samotným trénováním.

Apple veřejně neuvádí, jak je model VisionFeaturePrint navržen. Nejsou k dispozici informace o přesném počtu konvolučních a sdružovacích vrstev ani použitých aktivačních funkcích. Dostupné zdroje uvádí, že model byl vytrénován na otevřeném datasetu ImageNet, který obsahuje přes 14 milionů fotografií pro více než 20 tisíc kategorií [53; 54]. Jedná se však o informaci z roku 2019 a je více než pravděpodobné, že Apple své před-trénované modely v průběhu času vylepšuje.

Z výsledného grafu (Obrázek 30) lze vyčíst, že model dosáhl úspěšnosti 96 % během učení, 94 % během validace a 78 % během testování. Výsledek učení lze považovat za úspěšný a model je možné použít v aplikaci.



Obrázek 30 - Graf výsledku učení neuronové sítě (vlastní zpracování)

CreateML je inovativní nástroj, který výrazně zjednodušuje proces vytváření a exportu modelů umělých neuronových sítí pro vývojáře aplikací založených na jazyce Swift. Jedním z hlavních předností tohoto nástroje je jeho schopnost minimalizovat výslednou datovou velikost modelů, což zjednodušuje jejich nasazení na mobilních zařízeních. Vytvořený model má velikost zhruba 900 kB, a proto umožňuje rychlé nasazení na různé typy mobilních zařízení, bez znatelné zátěže na datové úložiště nebo výkon.

Poté, co se model importuje do projektu, je možné provést jeho inicializaci. Ta se provádí zavoláním konstruktoru třídy, která se jmenuje stejně jako použitý model, v tomto případě se jedná o třídu *FoodRecognitionModel* (Obrázek 31), kterou Swift sám na základě modelu vytvoří a provede jeho načtení.

```swift
class FoodRecognitionService: ObservableObject {
    @Published var model: VNCoreMLModel?

    @Published var results: [String] = []

    @Published var capturedImage: UIImage? = nil

    init() {
        let configuration = MLModelConfiguration )

        if let foodRecognitionModel = try? FoodRecognitionModel(configuration: configuration),
           let model = try? VNCoreMLModel(for: foodRecognitionModel.model) {
            self.model = model
        } else {
            model = nil
        }
    }

    func recognizeFood(image: CIImage) -> [String] {•••}
}
```

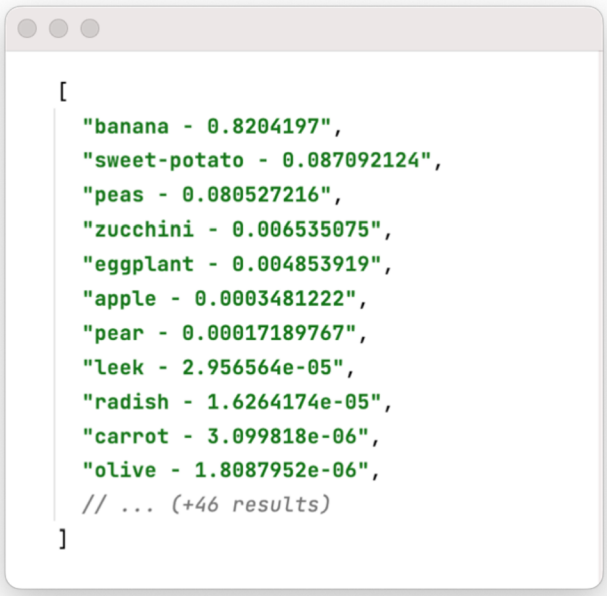
Obrázek 31 - Ukázka inicializace modelu v aplikaci (vlastní zpracování)

```swift
class FoodRecognitionService: ObservableObject {
    @Published var model: VNCoreMLModel?

    @Published var results: [String] = []

    @Published var capturedImage: UIImage? = nil

    init() {•••}

    func recognizeFood(image: CIImage) -> [String] {
        guard let model = model else {
            return []
        }

        let request = VNCoreMLRequest(model: model)

        let handler = VNImageRequestHandler(ciImage: image)

        try? handler.perform([request])

        guard let results = request.results as? [VNClassificationObservation] else {
            return []
        }

        let identifiers = Array(results.map { "\($0.identifier)" }[0 ... 3])

        return identifiers
    }
}
```

Obrázek 32 - Použití modelu pro detekci (vlastní zpracování)

Vstupní data ve formě fotografie pro model umělé neuronové sítě jsou získána stejným způsobem, jako v případě čtení čárového kódu, a to využitím frameworku AVFoundation a knihovny Vision (Obrázek 22). Voláním funkce *startCapturing* je získán aktuální snímek z kamery (Obrázek 23). Ten je dále odeslán do funkce *recognizeFood* (Obrázek 32). Je vytvořen objekt typu *VMImageRequestHandler*, který přijímá snímek jako parametr a poskytuje funkci *perform* s parametrem *VNCoreMLRequest*, který specifikuje požadovanou operaci. V tomto případě bude snímek vstupním parametrem pro definovanou umělou neuronovou síť.

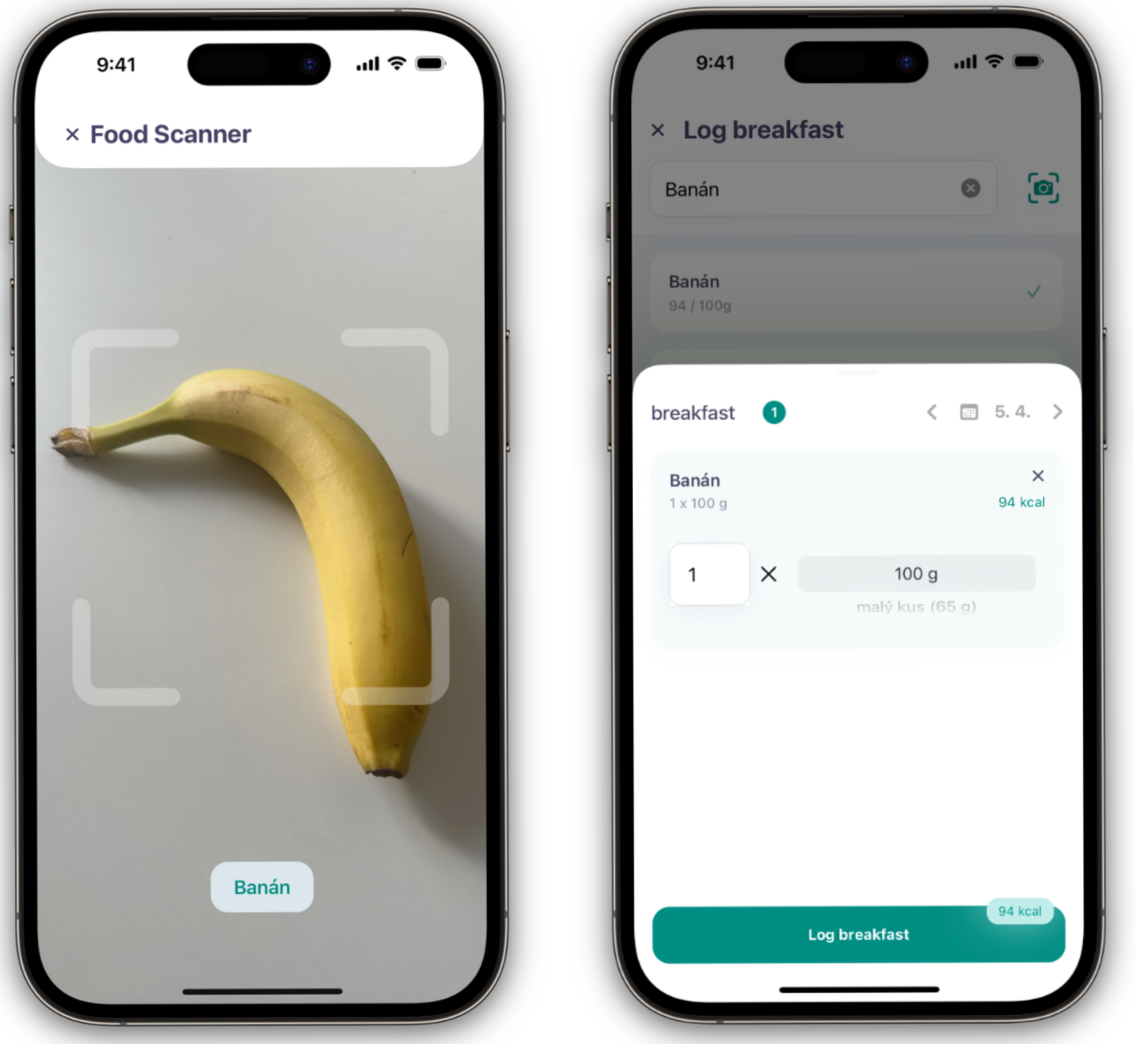
Výsledkem operace je seznam všech objektů, které umělá neuronová síť dokáže identifikovat, seřazený sestupně dle číselné hodnoty, která vyjadřuje pravděpodobnost, že se daný objekt na snímku nachází. Suma všech číselných hodnot je vždy rovna hodnotě 1 (Obrázek 33). Seznam je následně zkrácen, aby obsahoval pouze první tři výsledky dle nejvyšší hodnoty pravděpodobnosti.



Obrázek 33 - Výstupní hodnoty umělé neuronové sítě (vlastní zpracování)

V aplikaci je uživateli seznam zobrazen ve formě názvu potraviny. Po kliknutí na konkrétní název aplikace odešle požadavek na server, kde je provedeno vyhledávání v databázi potravin v tabulce *Foods* dle parametru *name* (Obrázek 12). Pokud je

potravina úspěšně nalezena, server pošle aplikaci odpověď ve formě záznamu potraviny. V aplikaci je název potraviny zobrazen uživateli s možností volby požadované varianty a zapsání do jídelníčku (Obrázek 34).



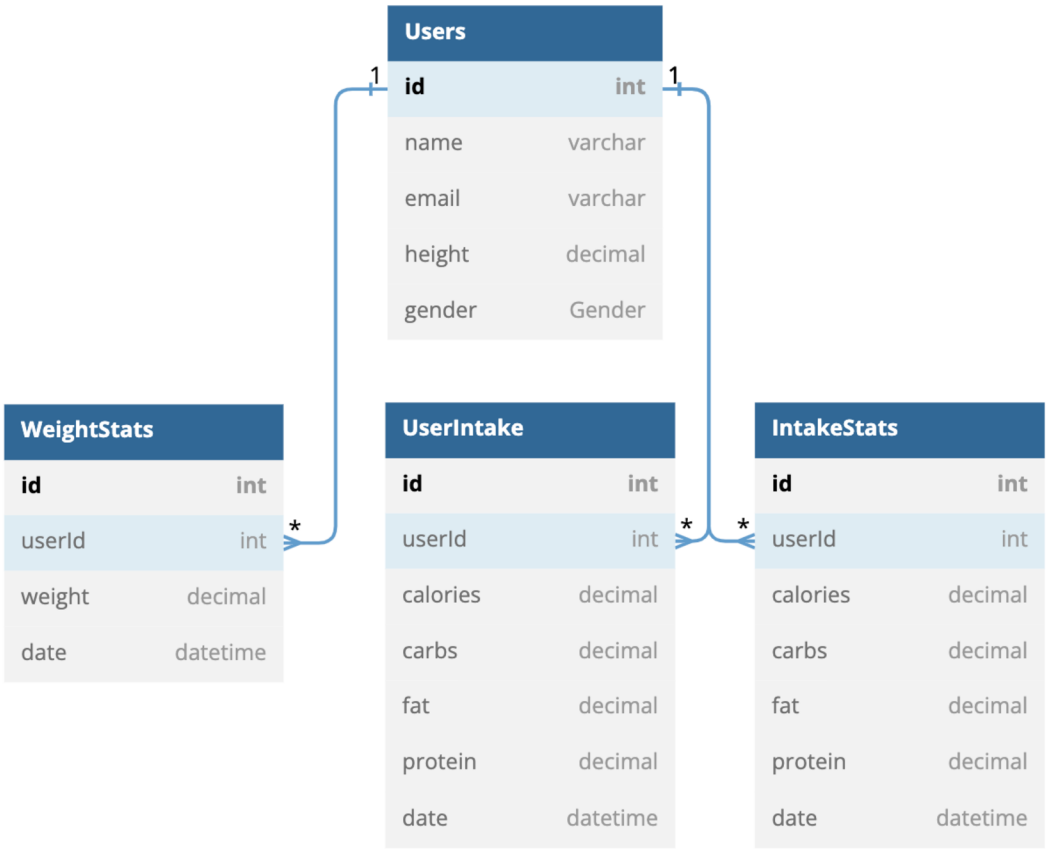
Obrázek 34 - Identifikace potraviny z fotografie (vlevo),
výběr potraviny pro zápis do jídelníčku (vpravo) (vlastní zpracování)

Funkce *recognizeFood* je automaticky volána každé dvě vteřiny. Tím je seznam identifikovaných potravin aktualizován dle aktuálního snímku z kamery. Rychlost samotné detekce na úrovni neuronové sítě byla měřena na základě provedení 100 po sobě jdoucích detekcí. Z naměřených hodnot činí průměrná rychlost detekce 5,8 ms na zařízení iPhone 14 Pro z roku 2022 s čipem Apple A16 Bionic a 72,5 ms na zařízení iPhone X z roku 2017 s čipem Apple A11 Bionic. Díky optimalizaci frameworku CoreML a součinnosti speciálních akceleračních čipů Neural Engine v čipech řady Bionic je možné dosáhnout vysokých rychlostí i na starších mobilních zařízeních.

8.5 Statistiky a doporučení

Sledování pokroku patří mezi jednu z nejdůležitějších funkcí aplikace. Pokud uživatel pravidelně zaznamenává svůj kalorický příjem, statistiky poskytují užitečný přehled o tom, jak se jeho úsilí promítá do reálných výsledků, což poskytuje dodatečnou motivaci v dosažení cílů. Statistiky poskytují objektivní pohled na výsledky dle uživatelských dat. Díky tomu je možné kalorický a výživový plán každému uživateli individuálně přizpůsobovat v průběhu času tak, aby byl cíl ambiciózní, ale zároveň realisticky dosažitelný.

Aplikace sleduje a zaznamenává denní hodnoty tělesné váhy, celkového příjmu kalorií a makroživin. Statistické hodnoty jsou organizovány v databázových tabulkách *WeightStats* (Tabulka 12) a *IntakeStats* (Tabulka 10) (Obrázek 35).



Obrázek 35 - Schéma databáze pro statistická data (vlastní zpracování)

Záznamy v tabulce *IntakeStats* jsou aktualizovány automaticky pomocí triggeru, který je spuštěn při jakékoliv změně v tabulce *Diaries*. Pokud například uživatel zapíše potravinu do jídelníčku, spuštěný trigger vypočítá příjem kalorií a makroživin pro konkrétní den a výslednými hodnotami aktualizuje záznamy v tabulce *IntakeStats*. Stejným způsobem dochází k aktualizaci hodnot v případě odstranění potraviny z jídelníčku nebo při úpravě množství. Díky atributu *date* je možné upravovat záznamy v minulosti a zároveň udržovat dlouhodobý přehled. Atributy jsou popsány v následující tabulce (Tabulka 10).

Tabulka 10 - Popis tabulky IntakeStats

Tabulka *IntakeStats*	
Atribut	**Význam**
id	Unikátní identifikátor
userId	Identifikátor uživatele
calories	Hodnota celkového denního příjmu kalorií
date	Datum vytvoření statistiky

Tabulka *UserInstake* udržuje přehled o plánech příjmu kalorií a makroživin uživatelů. Každý záznam představuje konkrétní výživový plán, který byl uživateli vypočítán na základě vyplnění vstupního dotazníku nebo úpravě dle statistických dat. V rámci uvedení konkrétního případu si lze představit nového uživatele, kterému aplikace vytvoří plán pro hubnutí s určitými hodnotami celkového příjmu kalorií, sacharidů, tuků a bílkovin. Plán je uložen do tabulky *UserIntake* a aplikace s ním dále pracuje. Po určitém čase dojde u uživatele ke zlepšení zdravotních výsledků v podobě poklesu váhy na cílenou hodnotu. V tuto chvíli je nutné vytvořit nový plán, který bude reflektovat aktuální potřeby uživatele. Plán je uložen do tabulky *UserIntake*. Aplikace dále pracuje pouze s tímto novým plánem a původní zůstává k dispozici pouze pro účely zobrazení historických statistických dat. Atributy jsou popsány v následující tabulce (Tabulka 11).

Tabulka 11 - Popis tabulky UserIntake

Tabulka *UserIntake*	
Atribut	**Význam**
id	Unikátní identifikátor
userId	Identifikátor uživatele
calories	Hodnota celkového denního příjmu kalorií
carbs	Hodnota celkového denního příjmu sacharidů
fat	Hodnota celkového denního příjmu tuků
protein	Hodnota celkového denního příjmu bílkovin
date	Datum vytvoření záznamu

Statistická data v tabulce *WeightStats* jsou aktualizována na základě vývoje tělesné váhy uživatele, který má možnost každý den zaznamenat její aktuální hodnotu. Díky tomu je možné společně s údaji z tabulek *IntakeStats* a *UserIntake* vypočítat a predikovat trend, kterým se hodnota tělesné váhy pohybuje.

Tabulka 12 - Popis tabulky WeightStats

Tabulka *WeightStats*	
Atribut	**Význam**
id	Unikátní identifikátor
userId	Identifikátor uživatele
weight	Hodnota tělesné váhy v kg
date	Datum vytvoření statistiky

Pokud je například cílem uživatele hubnout a vývoj váhy má dle statistik za dané období stoupající tendenci i přesto, že uživatel dodržuje denní kalorický příjem, je možné provést adekvátní úpravy v individuálním plánu, což znamená snížit denní příjem kalorií, aby se docílilo většího kalorického deficitu, který je pro hubnutí nezbytný. Nový plán je následně uložen do tabulky *UserIntake*. Alternativou pak může být navýšení fyzické aktivity.

Aplikace disponuje možností propojení s aplikací Apple Health. Jedná se o centrální místo pro shromažďování a analýzu zdravotních a pohybových dat uživatele. Apple Health integruje data z různých zdrojů, včetně fitness aplikací, chytrých vah, sportovních náramků a jiných nositelných zařízení. Vyvíjená aplikace dokáže číst data o fyzické aktivitě uživatele jako je počet kroků, spálené kalorie a četnost cvičení.

Přístup k datum je řízen skrze framework HealthKit, který je součástí platformy iOS. Inicializace probíhá vytvořením objektu *HKHealthStore* a voláním funkce *authorize* (Obrázek 37). Ta má definovaný seznam identifikátorů (*identifiers*), které se použijí pro udělení přístupu k datům uživatele (Obrázek 38). Aplikace využívá identifikátory pro počet kroků, spálených kalorií a minut cvičení. Pokud uživatel aplikaci povolí přístup k těmto údajům, probíhá získání aktuální hodnoty za konkrétní den (Obrázek 36). Dle zjištěných dat je možné individuálně upravit plán uživatele pro dosažení stanoveného cíle (Obrázek 39).

```
func getStepsCount(completetion: @escaping (Double) -> Void) {
    let stepCountQuery = HKStatisticsQuery(
        quantityType: HKQuantityType.quantityType(forIdentifier: .stepCount)!,
        quantitySamplePredicate: predicate,
        options: .cumulativeSum
    ) { _, result, _ in

        if let result = result,
           let sum = result.sumQuantity() {
            let stepCount = sum.doubleValue(for: HKUnit.count())
            completetion(stepCount)
        }
    }

    healthStore!.execute(stepCountQuery)
}
```

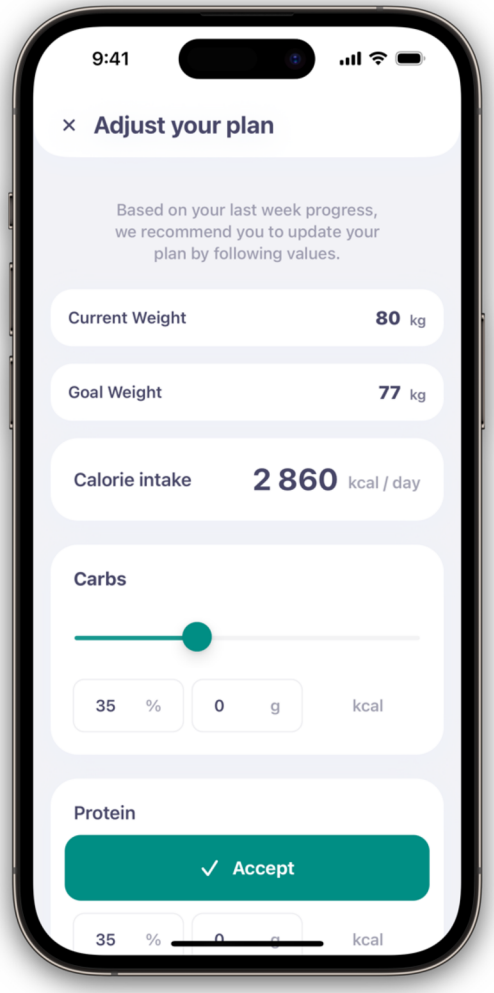
Obrázek 36 - Získání počtu kroků pro konkrétní den (vlastní zpracování)

```swift
class StatsService: ObservableObject {
    @Published var healthStore: HKHealthStore?

    @Published var predicate: NSPredicate?

    @Published var stepsCount: Double = 0
    @Published var activeEnergySum: Double = 0
    @Published var moveTimeSum: Double = 0

    init() {
        if HKHealthStore.isHealthDataAvailable() {
            healthStore = HKHealthStore()

            predicate = HKQuery.predicateForSamples(
                withStart: Calendar.current.startOfDay(for: Date()),
                end: .distantFuture,
                options: .strictEndDate)

            authorize { [self] result in
                if let _ = result {
                    getStepsCount { [self] count in
                        stepsCount = count
                    }
                    getActiveEnergySum { [self] sum in
                        activeEnergySum = sum
                    }
                    getMoveTimeSum { [self] sum in
                        moveTimeSum = sum
                    }
                }
            }
        }
    }

    func authorize(completetion: @escaping (Bool?) -> Void) {•••}

    func getStepsCount(completetion: @escaping (Double) -> Void) {•••}

    func getActiveEnergySum(completetion: @escaping (Double) -> Void) {•••}

    func getMoveTimeSum(completetion: @escaping (Double) -> Void) {•••}
}
```

Obrázek 37 - Integrace frameworku HealthKit (vlastní zpracování)

```swift
func authorize(completetion: @escaping (Bool?) -> Void) {
    let identifiers = Set([HKObjectType.quantityType(forIdentifier: .stepCount)!,
                           HKObjectType.quantityType(forIdentifier: .activeEnergyBurned)!,
                           HKObjectType.quantityType(forIdentifier: .appleExerciseTime)!])

    healthStore!.requestAuthorization(toShare: nil, read: identifiers) { success, _ in
        completetion(success)
    }
}
```

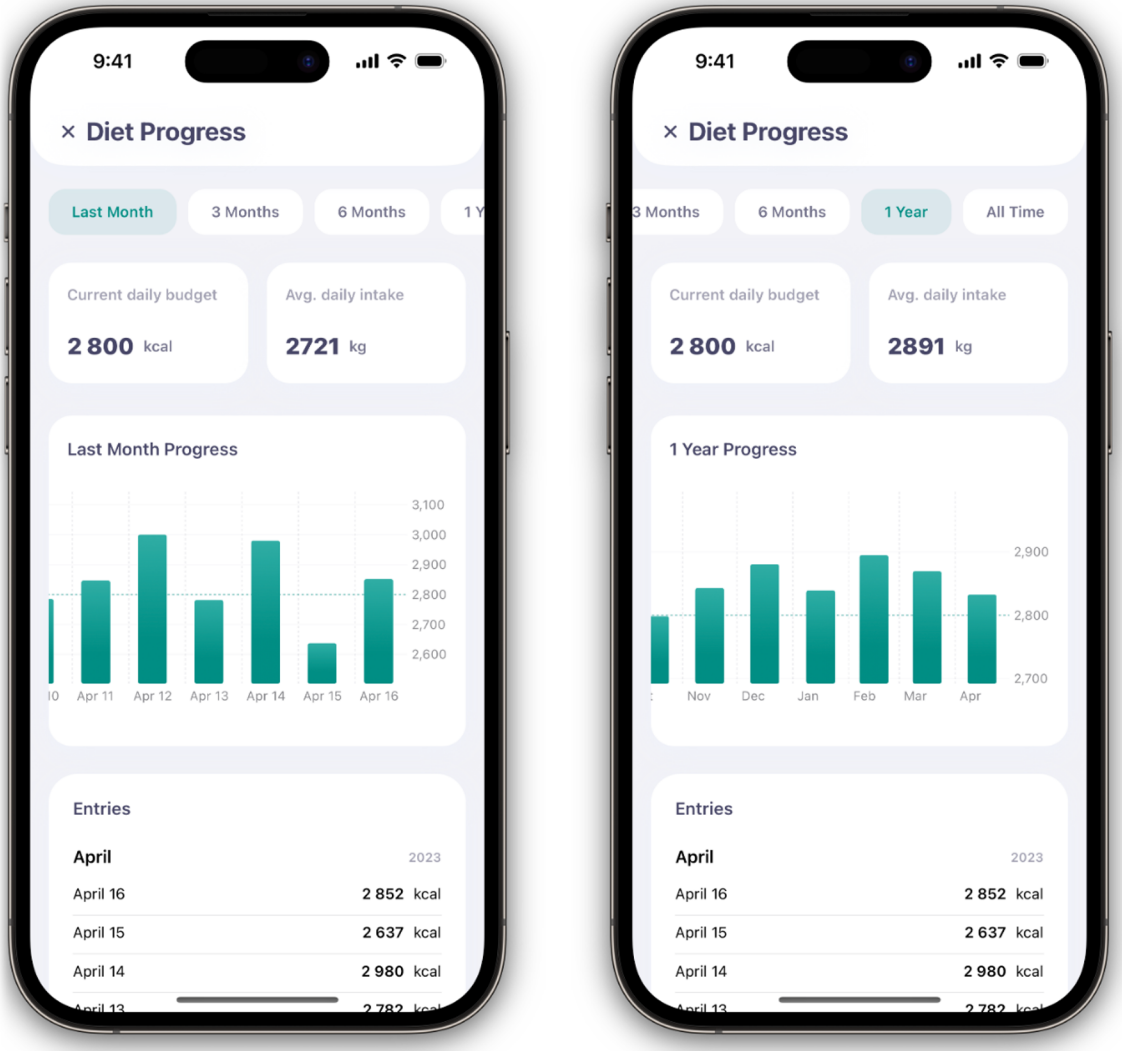
Obrázek 38 - Udělení povolení pro přístup k datům (vlastní zpracování)



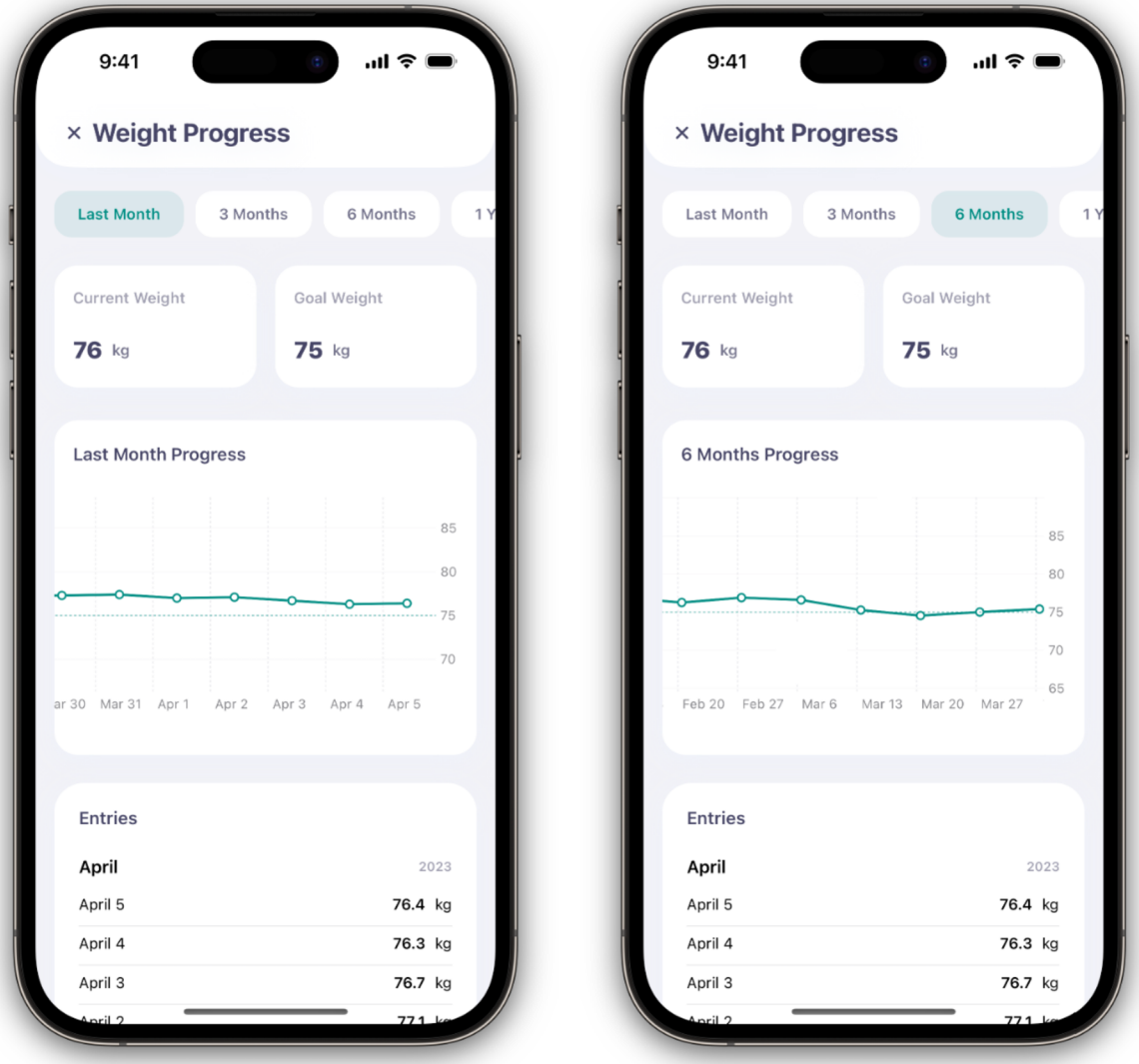
Obrázek 39 - Úprava plánu na základě statistik (vlastní zpracování)

Aplikace umožňuje zobrazení vývoje denního příjmu kalorií pomocí grafu, kde osa `X` reprezentuje datum a osa `Y` kalorickou hodnotu v kcal. Zobrazení je možné upravit výběrem časového rozmezí: Poslední měsíc, 3 měsíce, 6 měsíců, Rok a Vše (Obrázek 40). Data jsou načítána z tabulky *IntakeStats*.

Uživatel má také možnost zobrazit graf vývoje tělesné váhy. Osa `X` reprezentuje datum a osa `Y` hodnotu tělesné váhy v kg. Zobrazení je možné upravit podobně, jako v případě grafu vývoje kalorického příjmu, výběrem časového rozmezí: Poslední měsíc, 3 měsíce, 6 měsíců, Rok a Vše (Obrázek 41). Data jsou načítána z tabulky *WeightStats*.



Obrázek 40 – Statistiky vývoje denního kalorického příjmu (vlatsní zpracování)



Obrázek 41 - Statistiky vývoje tělesné váhy (vlastní zpracování)

9 Testování aplikace

Správná funkčnost aplikace byla ověřena řadou testovacích scénářů. Ty definují seznam kroků, které uživatel provede během používání aplikace, včetně možných vstupů a výstupů. Prvním krokem je identifikace cílové skupiny. Je důležité mít přehled o tom, kdo bude aplikaci používat a vytvořit scénář, který bude co nejpřesněji odpovídat reálným situacím, ve kterých se uživatelé ocitnou. Cílovou skupinou aplikace pro sledování kalorického příjmu jsou obvykle lidé, kteří chtějí kontrolovat svůj kalorický příjem za účelem hubnutí, udržení zdravé tělesné váhy nebo zlepšení své výkonnosti při sportovních aktivitách.

9.1 Individuální kalorický a výživový plán

Jednou z prvních funkcí aplikace, se kterou se uživatel setká, je úvodní vstupní dotazník. Jeho úkolem je vytvořit kalorický a výživový plán dle individuálních potřeb uživatele na základě vyplnění série jednoduchých otázek týkajících se zdraví, životního stylu a fyzické aktivity. V závislosti na pohlaví, věku, tělesné hmotnosti, výšce, úrovni fyzické aktivity a stanoveném cíli jsou uživatelům stanoveny hodnoty denního doporučeného příjmu kalorií a makroživin.

Tabulka 13 definuje scénář včetně vstupních a výstupních hodnot pro uživatele s cílem hubnout. Denní příjem kalorií pro hubnutí byl spočítán na 2 140 kcal. To představuje nižší hodnotu než 2 480 kcal pro udržení stávajícího fyzického stavu a vzniká tím kalorický deficit, který je pro hubnutí nezbytný. Příjem makroživin se pohybuje v doporučených hodnotách 30 % bílkoviny, 35 % sacharidy a 35 % tuky. Výsledný plán lze považovat za adekvátní a udržitelný.

Tabulka 13 - Testovací scénář pro plán hubnutí

Scénář s cílem hubnout	
Vstupní parametry	
Parametr	**Hodnota**
Pohlaví	Muž
Věk	25 let
Hmotnost	110 kg
Výška	179 cm
Fyzická aktivita	Sedavý způsob života
Cíl	Zhubnout
Výstupní parametry	
Parametr	**Hodnota**
Doporučený denní příjem kalorií	2 480 kcal
Denní příjem pro kalorický deficit	2 140 kcal
BMR	2 098 kcal
BMI	34.3 (obezita prvního stupně)
Sacharidy	187 g
Bílkoviny	160 g
Tuky	83 g

Dalším testovaným scénářem je uživatel s cílem nabrat svaly. Tabulka 14 definuje vstupní a výstupní hodnoty. Vypočítaný denní příjem kalorií je dle vstupního dotazníku 2828 kcal. Jedná se o vyšší hodnotu než 2 532 kcal pro zachování aktuální fyzického stavu, a tím pádem se jedná o kalorický surplus, který je současně se silovým cvičením nezbytný pro nabírání svalové hmoty. Příjem makroživin je rozdělen v doporučeném poměru 30 % bílkoviny, 35 % sacharidy a 35 % tuky. Plán příjmu kalorií a makroživin lze považovat za vhodný pro stanovený cíl.

Tabulka 14 - Testovací scénář pro plán nabírání váhy

Scénář s cílem nabrat svaly	
Vstupní parametry	
Parametr	**Hodnota**
Pohlaví	Muž
Věk	18 let
Hmotnost	65 kg
Výška	175 cm
Fyzická aktivita	Středně aktivní
Cíl	Nabrat svaly
Výstupní parametry	
Parametr	**Hodnota**
Doporučený denní příjem kalorií	2 532 kcal
Denní příjem pro kalorický nadbytek	2 828 kcal
BMR	1 658 kcal
BMI	21.2 (normální váha)
Sacharidy	247 g
Bílkoviny	212 g
Tuky	109 g

9.2 Vyhledávání a rozpoznání potravin

Aplikace nabízí řadu možností vyhledávání potravin. Nejběžnější je vyhledávání podle názvu potraviny. To probíhá skrze implementované fulltextové vyhledávání na straně serveru a databáze. Použitý český slovník provádí normalizaci slov na základní tvar. Hodnoty jsou následně porovnány s názvy potravin v databázi a je vypočítáno skóre relevantnosti a podobnosti. Podle hodnot skóre jsou výsledky seřazeny.

Během implementace bylo zjištěno, že fulltextové vyhledávání nefunguje správně, pokud je použit výchozí anglický slovník. Slova nejsou správně převedena na základní tvar a problém zároveň představuje česká diakritika. Bylo vytvořeno řešení v podobě odstranění diakritiky hledaného výrazu, nicméně bylo zapotřebí odstranit diakritiku i u jednotlivých potravin během porovnávání. To představovalo znatelnou zátěž na výpočetní výkon a vyhledávání bylo v tomto stavu pomalé. Dokončení dotazu běžně trvalo několik vteřin, a to i při použití indexů.

Následná implementace zahrnovala využití českého slovníku, kdy již nebylo nutné provádět manuální normalizaci slov. Vyhledávání se podařilo výrazně zrychlit a díky optimalizacím a použitým indexům se nyní rychlost pohybuje na hodnotě 6 ms.

Vyhledávání podle čárového kódu bylo implementováno pomocí frameworku AVFoundation a knihovny Vision. Proces vyhledávání je zahájen získáním aktuálního snímku z kamery zařízení. Ten je následně předán modelu umělé neuronové sítě z knihovny Vision, která se pokusí na snímku nalézt čárový kód a získat jeho textovou reprezentaci. V případě pozitivního nálezu se odešle požadavek na server, který provede vyhledávání dle v tabulce *Foods* dle parametru *ean*. V databázi není parametr *ean* vyplněn u všech záznamů. Pokud se stane, že čárový kód není nalezen, je uživateli zobrazena informativní zpráva.

Možnost vyhledávání podle fotografie potraviny byla navržena s použitím frameworku pro strojové učení CoreML. Byl vytvořen vlastní dataset obsahující 2 900 fotografií pro rozpoznávání 58 druhů ovoce a zeleniny. Umělá neuronová síť

trénovaná nad tímto datasetem dosáhla úspěšnosti 78 %. Použití této varianty vyhledávání se v praxi ukázalo jako užitečné pro rychlý zápis potravin do jídelníčku.

Úspěšnost rozpoznání na reálných fotografiích je ve většině případů velmi dobrá (Obrázek 42). Existují však druhy ovoce a zeleniny, které si jsou vzájemně podobné a v krajních případech je umělá neuronová síť zamění. Ukázkový příklad představuje rajče a jablko, případně pomeranč a mandarinka (Obrázek 43). Problém lze řešit zobrazením prvních tří výsledků podle nejvyšší hodnoty, která vyjadřuje pravděpodobnost, že se daný objekt na snímku nachází. Místo aktuálního jednoho výsledku, který může být nepřesný, by měl uživatel možnost výběru z několika možných kandidátů. Další variantou je vytvoření nového robustnějšího datasetu. Použití většího množství kvalitnějších fotografií jednotlivých druhů ovoce a zeleniny může zvýšit přesnost neuronové sítě během učení, a tím také zlepšit výslednou úspěšnost detekce.

Obrázek 42 - Ukázka správného rozpoznání ovoce (vlastní zpracování)

Obrázek 43 - Ukázka správného a chybného rozpoznání (vlastní zpracování)

10 Shrnutí výsledků

V rámci této diplomové práce proběhl průzkum moderních dostupných technologií z oblasti mobilních aplikací, umělých neuronových sítí a počítačového vidění s cílem vytvořit mobilní aplikaci na platformě iOS pro sledování kalorického příjmu.

Došlo k vytvoření mobilní aplikace, jejíž funkcionalita odpovídá stanoveným cílům a byly implementovány všechny funkční požadavky dle kapitoly 7.3 Funkční požadavky.

Pro správu a organizaci dat byla navržena databáze v relačním databázovém systému PostgreSQL, včetně fulltextového vyhledávání (s využitím českého slovníku) v tabulce *Foods* nad atributem *name*, aby bylo možné vyhledávat potraviny dle názvu. Na rozdíl od pouhého porovnávání slov je fulltextové vyhledávání přesnější a komplexnější. Vyhledává v kompletním textovém dokumentu a umí pracovat s výrazovými vztahy mezi slovy. Fulltextové vyhledávání je jazykově závislé. Díky optimalizacím a použitým indexům je rychlost provedení dotazu zhruba 6 ms.

Jednou z hlavních funkcí je vlastní model umělé neuronové sítě, který slouží pro identifikaci potraviny dle pořízeného snímku z kamery mobilního zařízení. V rámci ověření proveditelnosti je navržený model určen k rozpoznání 58 druhů ovoce a zeleniny. Dataset je tvořen 2 900 fotografiemi a model dosahuje úspěšnosti 78 %. V kapitole 9 Testování aplikace byly popsány krajní případy, kdy rozpoznání potraviny nefunguje správně a dochází k záměnám, například rajče a jablko nebo pomeranč a mandarinka. Kapitola také zmiňuje možné způsoby řešení tohoto problému.

Model má velikost zhruba 900 kB, a proto umožňuje rychlé nasazení na různé typy mobilních zařízení, bez znatelné zátěže na datové úložiště nebo výkon. Výhodou je nezávislost na internetovém připojení. Umělá neuronová síť je provozována přímo na mobilním zařízení, čímž je zvýšen důraz na soukromí uživatelských dat. Rychlost neuronové sítě byla popsána na závěr kapitoly 8.4 Rozpoznání potraviny z fotografie. Rychlost na úrovni neuronové sítě byla měřena na základě provedení

100 po sobě jdoucích detekcí. Z naměřených hodnot činí průměrná rychlost detekce 5,8 ms na zařízení iPhone 14 Pro z roku 2022 s čipem Apple A16 Bionic a 72,5 ms na zařízení iPhone X z roku 2017 s čipem Apple A11 Bionic. Díky optimalizaci frameworku CoreML a součinnosti speciálních akceleračních čipů Neural Engine v čipech řady Bionic je možné dosáhnout vysokých rychlostí i na starších mobilních zařízeních.

Navržený vstupní dotazník umožňuje na základě série otázek týkajících se zdraví, životního stylu, fyzické aktivity a výživových preferencí vytvořit plán příjmu kalorií a makroživin pro každého uživatele na základě jeho individuálních potřeb. Průchod vstupním dotazníkem a výpočet všech potřebných hodnot jsou popsány v kapitole 8.2 Sledování kalorického příjmu.

11 Závěr

Cílem diplomové práce bylo vytvořit mobilní aplikaci na platformě iOS pro sledování kalorického příjmu a podpoření zdravého životního stylu s využitím dostupných moderních technologií z oblasti mobilních aplikací, umělých neuronových sítí a počítačového vidění.

Vývoj aplikace zahrnoval analýzu a rozbor stávajících dostupných aplikací na trhu pro správnou identifikaci cílové skupiny uživatelů. Tou jsou lidé, kteří chtějí kontrolovat svůj kalorický příjem za účelem hubnutí, udržení zdravé tělesné váhy a kondice nebo zlepšení své výkonnosti během sportovních aktivit.

Stanovený cíl se podařilo splnit včetně všech funkčních požadavků, dle kapitoly 7.3 Funkční požadavky.

Vytvořená databáze potravin v relačním databázovém systému PostgreSQL umožňuje fulltextové vyhledávání potravin dle názvu. Implementace zahrnuje využití českého slovníku pro normalizaci slov a porozumění jazyka pro získání sémanticky smysluplných výsledků. Díky řadě optimalizacím a použitým indexům je dosahuje rychlost dotazu 6 ms.

Vstupní dotazník umožňuje vytvořit plán příjmu kalorií a makroživin pro každého uživatele dle individuálních potřeb na základě série otázek týkajících se zdraví, životního stylu, fyzické aktivity a výživových preferencí.

Klíčovou funkcí aplikace je umělá neuronová síť, která slouží pro rozpoznání a identifikaci potravin dle pořízeného snímku z kamery zařízení. Vytvořený model neuronové sítě se jeví jako velmi dobře použitelný. Aktuální implementace dosahuje úspěšnosti detekce 78 % a dokáže rozpoznat celkem 58 druhů ovoce a zeleniny. V případě dalšího vývoje aplikace je možné dataset umělé neuronové sítě rozšířit o další druhy potravin, například pečivo nebo masné výrobky. Velikost modelu je 900 kB, a proto umožňuje rychlé nasazení na různé typy mobilních zařízení, bez znatelné zátěže na datové úložiště nebo výkon.

V případě dalšího vývoje je možné rozšířit sledování makroživin i o hodnoty vlákniny, soli, cukru nebo nasycených a nenasycených tuků, a to přidáním atributů do databázové tabulky *Foods*. Stejným způsobem je možné provést rozšíření o hodnoty obsahu minerálních látek, vitamínů a aditivních potravinářských látek (tzv. éčka).

Vzhledem k tomu, že serverová část aplikace je implementována frameworkem Next.js, který se používá nejen pro tvorbu serverových aplikací s využitím architektury webových služeb, ale také k tvorbě interaktivních uživatelských rozhraní, je možné v případě dalšího vývoje implementovat webovou aplikaci. Ta by fungovala podobně jako mobilní aplikace, pouze bez možnosti vyhledávání potravin podle čárového kódu nebo rozpoznání ze snímku z kamery. Webová aplikace by využívala stejné rozhraní API jako mobilní aplikace. Webová aplikace nebyla cílem této práce, nicméně díky zvolené architektuře je možné webovou aplikaci vytvořit bez změn ve stávající implementaci.

Seznam použité literatury

1. **czso.cz.** Jak jsou na tom Češi s chudobou, obezitou či sportováním? [Online] Duben 2018. https://www.czso.cz/csu/stoletistatistiky/jak-jsou-na-tom-cesi-s-chudobou-obezitou-ci-sportovanim.

2. **V. Osilla, Eva, O. Safadi, Anthony a Sharma, Sandeep.** Calories. [Online] Září 2022. https://www.ncbi.nlm.nih.gov/books/NBK499909/.

3. **Global BMI Mortality Collaboration.** Body-mass index and all-cause mortality. [Online] Srpen 2016. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4995441/.

4. **Pethusamy, Karthikeyan, Gupta , Anshul a Yadav, Rahul.** Basal Metabolic Rate (BMR). [Online] Bžezen 2019. https://link.springer.com/referenceworkentry/10.1007/978-3-319-47829-6_1429-1.

5. **Frothingham, Scott.** healthline.com. [Online] Prosinec 2018. https://www.healthline.com/health/what-is-basal-metabolic-rate.

6. **ncbi.nlm.nih.gov.** A Biometric Study of Human Basal Metabolism. [Online] Prosinec 1918. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1091498/.

7. **Frankenfield, David, Roth-Yousey, Lori a Compher, Charlene.** Comparison of Predictive Equations for Resting Metabolic Rate in Healthy Nonobese and Obese Adults: A Systematic Review. [Online] Květen 2005. https://www.sciencedirect.com/science/article/abs/pii/S0002822305001495.

8. **Biddulph, Maddy.** livescience.com. [Online] Červenec 2022. https://www.livescience.com/what-is-a-calorie-deficit.

9. **Deen, JC.** pcyc.org.au. [Online] Březen 2020. https://www.pcyc.org.au/blog/nutrition-tip-what-is-a-caloric-surplus/.

10. **National Academy of Sciences.** Recommended Dietary Allowances. [Online] Leden 1989. https://www.ncbi.nlm.nih.gov/books/NBK234938/.

11. **Venn, Bernard J.** Macronutrients and Human Health for the 21st Century. [Online] Srpen 2020. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7468865/.

12. **Watford, Malcolm.** ncbi.nlm.nih.gov. [Online] Červenec 2018. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6140426/.

13. **Slavin, Joanne.** ncbi.nlm.nih.gov. [Online] Listopad 2014. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4224210/.

14. **Ahmed, Saba.** ncbi.nlm.nih.gov. [Online] Květen 2022. https://www.ncbi.nlm.nih.gov/books/NBK525952/.

15. **Tomanka, Marek.** Řekni mi, co jíš. [Online] Duben 2022. https://forbes.cz/vyzivove-hodnoty-lide-s-nami-uz-zhubli-pres-tisic-tun-rika-sef-kalorickych-tabulek/.

16. **kalorickетabulky.cz.** Co jsou Kalorické Tabulky. [Online] Leden 2023. https://www.kalorickетabulky.cz/manual.

17. **kalorickетabulky.cz**. Kalorické tabulky Premium. [Online] Leden 2023. https://www.kalorickетabulky.cz/user/premium.

18. **macronaut.cz.** Macronaut. [Online] Leden 2023. https://macronaut.cz/#jak-to-funguje.

19. **macronaut.cz**. Macronaut. [Online] Leden 2023. https://macronaut.cz/price/.

20. **loseit.com.** Lose It! How it works. [Online] Leden 2023. https://www.loseit.com/how-it-works/.

21. **deepai.org.** Neuron. [Online] Květen 2019. https://deepai.org/machine-learning-glossary-and-terms/neuron.

22. **Ing. Ph.D. Blaha, Milan.** *Úvod do neuronových sítí - Biologická analogie.* [Online] Srpen 2020. https://portal.matematickabiologie.cz/index.php?pg=analyza-a-hodnoceni-biologickych-dat--umela-inteligence--neuronove-site-jednotlivy-neuron--uvod-do-neuronovych-siti--biologicka-analogie.

23. **Durčák, Ing. Pavel.** Neuronové sítě a princip jejich fungování. [Online] Září 2017. https://www.napocitaci.cz/33/neuronove-site-a-%20princip-jejich-fungovani-uniqueidgOkE4NvrWuNY54vrLeM670eFNQh552VdDDulZX7UDBY/.

24. **matematickabiologie.cz.** Biologická analogie. [Online] Leden 2020. https://portal.matematickabiologie.cz/index.php?pg=analyza-a-hodnoceni-biologickych-dat--umela-inteligence--neuronove-site-jednotlivy-neuron--uvod-do-neuronovych-siti--biologicka-analogie.

25. **Roy, Abhijit.** An Introduction to Gradient Descent and Backpropagation. [Online] Červen 2020. https://towardsdatascience.com/an-introduction-to-gradient-descent-and-backpropagation-81648bdb19b2.

26. **Cummins, N. Schraudolph a F.** Artificial Neuron Models. [Online] Leden 2006. https://cnl.salk.edu/~schraudo/teach/NNcourse/ann-overview.html.

27. **Institut biostatistiky a analýz Lékařské fakulty Masarykovy univerzity.** Koncept umělé neuronové sítě. [Online] Srpen 2020. https://portal.matematickabiologie.cz/index.php?pg=analyza-a-hodnoceni-biologickych-dat--umela-inteligence--neuronove-site-jednotlivy-neuron--uvod-do-neuronovych-siti--koncept-umele-neuronove-site.

28. **Knowledge Transfer.** Explain Pooling layers. [Online] Srpen 2021. https://androidkt.com/explain-pooling-layers-max-pooling-average-pooling-global-average-pooling-and-global-max-pooling/.

29. **Mgr. Martin Pilát, Ph.D.** Neuronové sítě - konvoluční sítě a zpracování obrazu. [Online] Prosinec 2021. https://martinpilat.com/cs/prirodou-inspirovane-algoritmy/neuronove-site-konvolucni-site-zpracovani-obrazu.

30. **apple.com.** Swift. [Online] Leden 2023. https://developer.apple.com/swift/.

31. **apple.com**. SwiftUI. [Online] Leden 2023. https://developer.apple.com/xcode/swiftui/.

32. **Herbert, David.** What is React.js. [Online] Červen 2022. https://blog.hubspot.com/website/react-js.

33. **Máca, Jindřich.** Úvod do ReactJS. [Online] Únor 2019. https://www.itnetwork.cz/javascript/react/zaklady/uvod-do-react/.

34. **nextjs.org.** What is Next.js. [Online] Leden 2023. https://nextjs.org/learn/foundations/about-nextjs/what-is-nextjs.

35. **Kvapil, Jiří.** Úvod do TypeScriptu. [Online] Květen 2018. https://www.itnetwork.cz/javascript/typescript/uvod-do-typescriptu.

36. **Lutkevich, Ben.** Application programming interface . [Online] Prosinec 2022. https://www.techtarget.com/searchapparchitecture/definition/application-program-interface-API.

37. **Salunke, Devendra.** RESTful Web Services. [Online] Prosinec 2022. https://www.geeksforgeeks.org/restful-web-services/.

38. **apple.com.** CoreML. [Online] Leden 2023. https://developer.apple.com/documentation/coreml.

39. **postgresql.org.** What is PostgreSQL. [Online] Leden 2023. https://www.postgresql.org/about/.

40. **Terra, John.** What is Client-Server Architecture. [Online] Říjen 2022. https://www.simplilearn.com/what-is-client-server-architecture-article.

41. **intellipaat.** What is Client Server Architecture. [Online] Prosinec 2022. https://intellipaat.com/blog/what-is-client-server-architecture/.

42. **Gillis, Alexander.** REST API. [Online] Září 2020. https://www.techtarget.com/searchapparchitecture/definition/RESTful-API.

43. **Open Food Facts.** Terms of use, contribution and re-use. *OpenFoodFacts.org.* [Online] 2023. https://world.openfoodfacts.org/terms-of-use.

44. **Ústav zemědělské ekonomiky a informací.** Databáze složení potravin ČR . *NutriDatabaze.cz.* [Online] 2020. https://www.nutridatabaze.cz.

45. **Walle, Gavin.** The Best Macronutrient Ratio for Weight Loss. [Online] Září 2018. https://www.healthline.com/nutrition/best-macronutrient-ratio#macro-ratio.

46. **postgresql.org.** Text Search Types. [Online] Březen 2023. https://www.postgresql.org/docs/current/datatype-textsearch.html.

47. **postgresql.org**. Full Text Search. [Online] Březen 2023. https://www.postgresql.org/docs/15/textsearch-intro.html#TEXTSEARCH-MATCHING.

48. **postgresql.org**. Controlling Text Search. [Online] Březen 2023. https://www.postgresql.org/docs/15/textsearch-controls.html#TEXTSEARCH-RANKING.

49. **postgresql.org**. *Functions and Operators pg_trgm.* [Online] Březen 2023. https://www.postgresql.org/docs/current/pgtrgm.html#PGTRGM-OP-TABLE.

50. **Lipscombe, Trevor.** *Ideal Body Weight Formulas in Relation to Body Mass Index and Reference Heights.* [Online] Březen 2021. https://www.academia.edu/45653943/Ideal_Body_Weight_Formulas_in_Relation_to_Body_Mass_Index_and_Reference_Heights.

51. **Apple.** Vision. [Online] Červen, 2022. https://developer.apple.com/documentation/vision.

52. **Chugh, Anupam.** Vision Image Similarity Using Feature Prints in iOS. [Online] Prosinec 2019. https://betterprogramming.pub/compute-image-similarity-using-computer-vision-in-ios-75b4dcdd095f .

53. **Singh, Navdeep.** Build iOS-ready machine learning models using Create ML. [Online] https://heartbeat.comet.ml/build-ios-ready-machine-learning-models-using-create-ml-cf35091f6f8c.

54. **Stanford Vision Lab.** Image Net. [Online] Leden 2020. https://www.image-net.org/.

55. **seldon.io.** Neural Network Models Explained. [Online] Leden 2022. https://www.seldon.io/neural-network-models-explained.

56. **oracle.com.** What Is a Database. [Online] Leden 2023. https://www.oracle.com/database/what-is-database/.

57. **apple.com.** CreateML. [Online] Leden 2023. https://developer.apple.com/machine-learning/create-ml/.

58. **Shubel, Meredith.** What Is TypeScript. [Online] Červenec 2022. https://thenewstack.io/what-is-typescript/.

Přílohy

Příloha č. 1

Zadání diplomové práce.

Univerzita Hradec Králové
Fakulta informatiky a managementu

Zadání diplomové práce

Autor:	**Bc. Daniel Krejčí**
Studium:	I2100067
Studijní program:	N1802 Aplikovaná informatika
Studijní obor:	Aplikovaná informatika
Název diplomové práce:	**Aplikace pro sledování kalorického příjmu na platformě iOS**
Název diplomové práce AJ:	Calorie tracking application for iOS platform

Cíl, metody, literatura, předpoklady:

Cíl: Prozkoumat problematiku a principy sledování kalorického příjmu a implementovat aplikaci s využitím moderních technologií, která tyto principy uplatňuje.

1) Úvod

2) Prozkoumání problematiky sledování kalorického příjmu, analýza aktuálních řešení na trhu

3) Prozkoumání využití moderních technologií, včetně neuronových sítí, pro sledování kalorického příjmu

4) Návrh implementace aplikace pro sledování kalorického příjmu

5) Implementace a testování navrhovaného řešení

6) Závěr a zhodnocení dosažených výsledků

Zadávající pracoviště:	Katedra informatiky a kvantitativních metod, Fakulta informatiky a managementu
Vedoucí práce:	doc. Ing. Filip Malý, Ph.D.

Datum zadání závěrečné práce: 26.1.2021