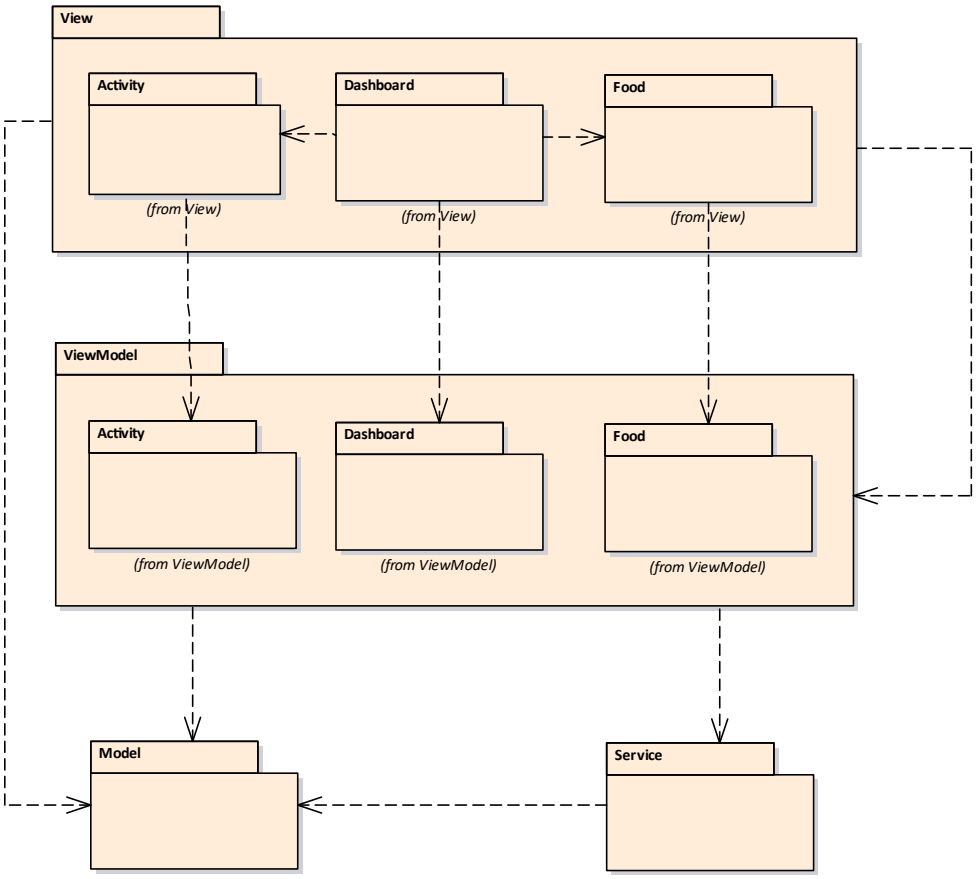
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ ČVUT V PRAZE

# Obsah

# 1. Návrh logické architektury (Aplikace)




Obrázek 1 - Návrh logické architektury (Aplikace)

**Aplikace** je druhou ze dvou hlavních součástí projektu.
Architektura aplikace je navržena jako **dvouvrstvá**. První, **datová - business** vrstva obsahuje **Model** s definicemi jednotlivých **DTO** (data transfer object), které slouží jako rozhraní k přenosu dat mezi Rest API, na které je aplikace napojená, a prezentační vrstvou. Dále tato vrstva obsahuje **Service** s definicemi a implementacemi jednotlivých rozhraní sloužících ke komunikaci s RestAPI, přihlašování uživatele a dalším logickým funkčnostem. Druhá, **prezentační** vrstva obsahuje **ViewModel**, který slouží k přenosu dat mezi datovo-business vrstvou a **View**, které zobrazuje data na obrazovku.
Pro implementaci aplikace byl využit framework **SwiftUI**, který usnadňuje vytváření a zobrazování grafického rozhraní.
Aplikace je psána v jazyce **Swift**, který je v dnešní době hlavním používaným jazykem pro vývoj iOS aplikací.
Použité knihovny a frameworky:

- Swift (https://swift.org/)
- SwiftUI (https://developer.apple.com/xcode/swiftui/)

## 1.1 Model

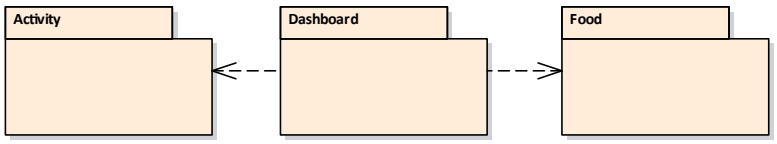
Balíček obsahuje datové třídy, které slouží jako rozhraní pro prezentační vrstvu, a které jsou načítány z Rest API pomocí Services. Jedná se o první část **datové-business** vrstvy.

## 1.2 Service

Balíček je určen pro rozhraní, která slouží ke komunikaci s Rest API a správě přihlášeného uživatele. Jednotlivé implementace těchto rozhraní budou také v tomto balíčku. Jedná se o druhou část **datové-business** vrstvy.

## 1.3 View

Balíček je určen pro třídy, které zobrazují jednotlivá data z ViewModelu uživateli pomocí grafického rozhraní. Jedná se o jednu ze dvou částí **prezentační** vrstvy.




Obrázek 2 - View

### 1.3.1 Activity

Balíček obsahuje views týkající se přidávání, úpravy a hledání aktivit.

### 1.3.2 Dashboard

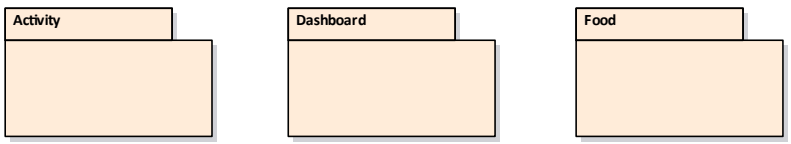
Balíček obsahuje views týkající se denního přehledu.

### 1.3.3 Food

Balíček obsahuje obrazovky týkající se přidávání, úpravy a hledání jídel.

## 1.4 ViewModel

Balíček je určen pro třídy, které načítají data z datovo-business vrstvy pomocí rozhraní a poskytují je jako pozorovatelné kolekce třídám z balíčku View. Jedná se o druhou část **prezentační** vrstvy.




Obrázek 3 - ViewModel

### 1.4.1 Activity

Balíček obsahuje view modely týkající se přidávání, úpravy a hledání aktivit.

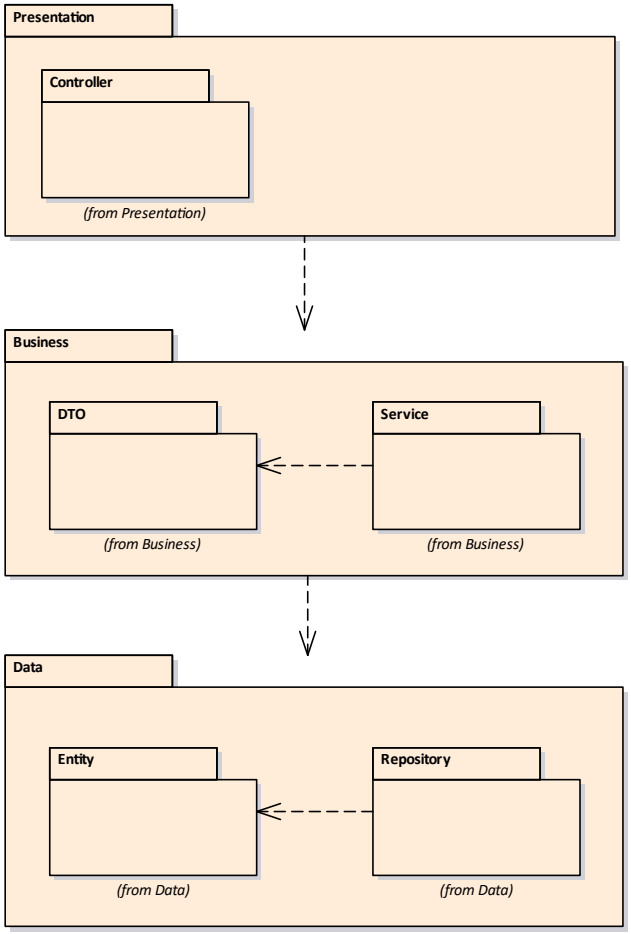
### 1.4.2 Dashboard

Balíček obsahuje view modely týkající se denního přehledu.

### 1.4.3 Food

Balíček obsahuje view modely týkající se přidávání, úpravy a hledání jídel.

# 2. Návrh logické architektury (RestAPI)




Obrázek 4 - Návrh logické architektury (RestAPI)

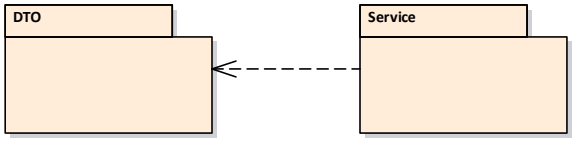
**Rest API** je jednou ze dvou hlavních součástí projektu.
Architektura Rest API je navržena jako **třívrstvá**. První, **datová** vrstva, obsahuje definici **entit** a **repository**. Druhá, **business** vrstva pracuje pomocí rozhraní **repository** s jednotlivými entitami a vytváří z nich **DTO**. Třetí, prezentační vrstva obsahuje **controllery**, které poskytují mapování HTTP požadavků na jednotlivé services.
Pro implementaci Rest API byl využit framework **Spring**, který usnadňuje vývoj enterprise aplikací, především pomocí **IoC** (Inversion of Control), které umožňuje provázání komponent pomocí **anotací**, čímž usnadňuje výměnu jednotlivých komponent za druhé.
Pro perzistenci dat je využit framework **Hibernate**, který staví nad rozhraním **JPA** (Java persistence api) - nezáleží tak na konkrétní databázi, ale pouze na definici entit. Typ použité databáze se tak nastavuje až v konfiguračním souboru.
Celé Rest API je psáno v jazyce **Kotlin**, který funguje jako jazyková nadstavba nad Javou. Pomocí různých zkratek a funkcionálních zápisů tak umožňuje vyhnutí se opakování stereotypních částí kódů, jako jsou například gettery, settery, toString a hashCode v POJO (data class).
Použité knihovny a frameworky:

- Java jazyková nadstavba - Kotlin (https://kotlinlang.org/)
- Provázání balíčků/komponent (IoC) - Spring (https://spring.io/)
- Persistence dat (ORM) - Hibernate (http://hibernate.org/)

- Správa buildů - Maven (https://maven.apache.org/)

## 2.1 Business

Veškerá business logika ze zadání bude implementována v balíčcích v této vrstvě. Vrstva dále komunikuje s prezenční vrstvou.




Obrázek 5 - Business

### 2.1.1 DTO

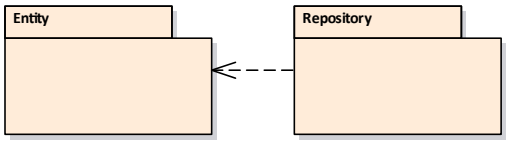
Balíček obsahuje všechny třídy, které slouží ke komunikaci business vrstvy s prezentační vrstvou. Zavedení DTO nad standardními entitami umožňuje prevenci zacyklení (objekty jsou nahrazeny ID) a také provádění dotazů, které nezávisí na konkrétních entitách.

### 2.1.2 Service

Balíček je určen pro třídy, které přijmou a zpracují z prezentační vrstvy jednotlivé požadavky. Součástí zpracování požadavků je také čtení, zapisování a úprava dat v databázi a práce s repozitáři, která probíhá přes rozhraní.

## 2.2 Data

Tato vrstva obsahuje třídy, které umožňují načítání, úpravu a ukládání dat z databáze.




Obrázek 6 - Data

### 2.2.1 Entity

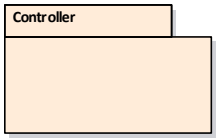
Balíček obsahuje definici datových objektů, se kterými Services pracují. Tento balíček bude využíván v business vrstvě pro ukládání do databáze.

### 2.2.2 Repository

Balíček obsahuje definici rozhraní, přes které bude business vrstva ukládat, načítat a upravovat data. Implementace rozhraní v tomto balíčku bude vytvořena automaticky pomocí Hibernate a JPA.

## 2.3 Presentation

Prezentační vrstva Rest API, která reaguje na HTTP požadavky a odesílá HTTP odpovědi. Tyto požadavky jsou pomocí Spring anotací převedeny do metod jednotlivých Controllerů, které následně komunikují se Services v business vrstvě.



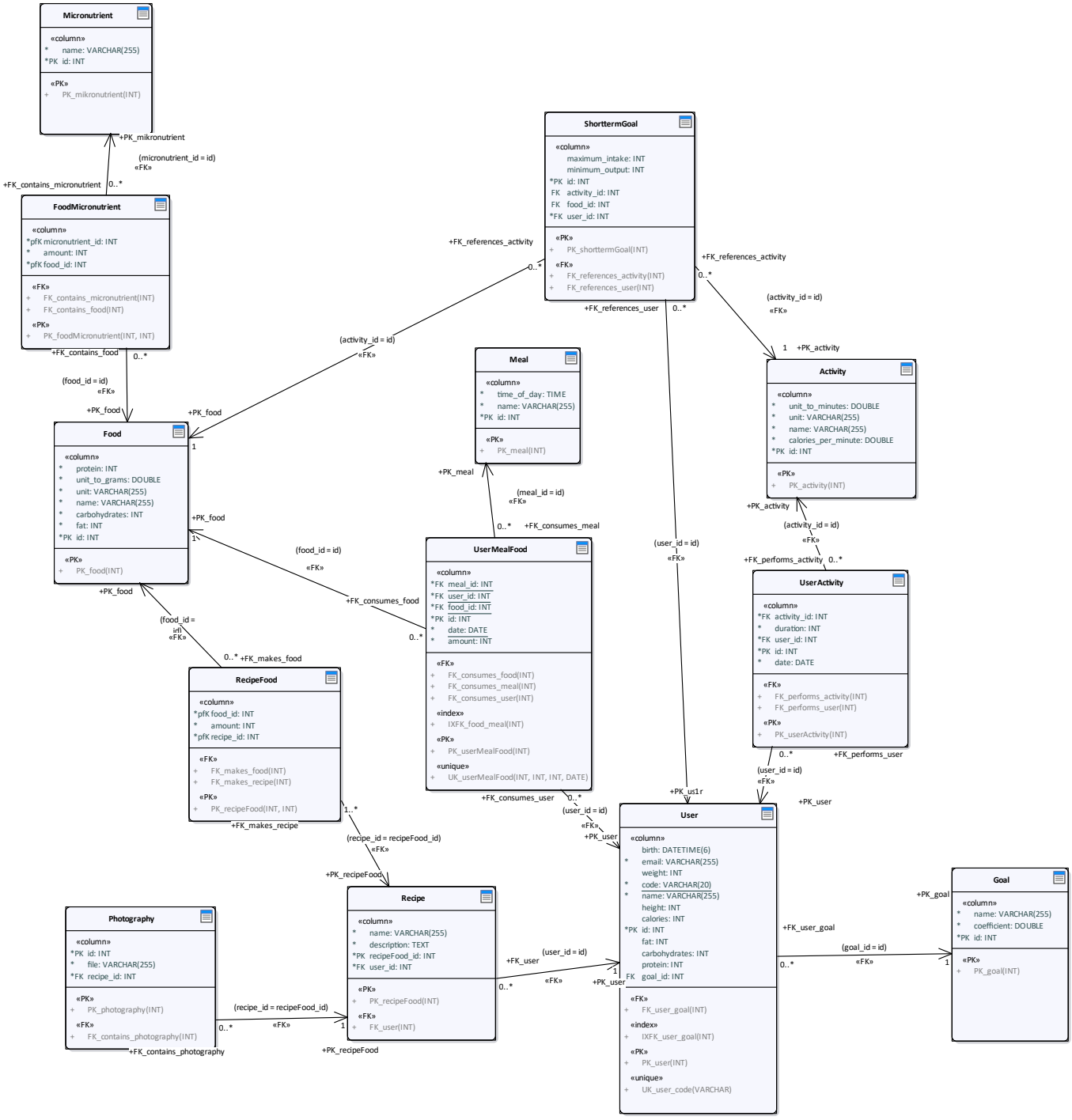

Obrázek 7 - Presentation

### 2.3.1 Controller

Balíček je určen pro třídy, které zpracují HTTP požadavky a transformují je na volání metod business vrstvy.

# 3. Databázový model

## 3.1 DDL




Obrázek 8 - DDL

### 3.1.1 Activity *«table»*

Konkrétna aktivita, ktorú môže zákazník vykonať.

| Název atributu | Datový typ | Not null | Popis |
|---|---|---|---|
| unit_to_minutes | DOUBLE | True | Převod jednotky, ve které se čas výkonu aktivity zadává, na minuty (1 ujitý kilometr = 10 minut) |
| unit | VARCHAR(255) | True | Jednotka, ve které se zapisuje výkon aktivity (metry, hodiny, minuty) |
| name | VARCHAR(255) | True | Název aktivity, resp. o akú aktivitu sa jedná. |
| calories_per_minute | DOUBLE | True | Výdej kalorií na 1 kg váhy při vykonávání aktivity po dobu 1 minuty |
| id | INT | True | |

### 3.1.2 Food *«table»*

Konkrétní jídlo, z kterého se skládá chod.

| Název atributu | Datový typ | Not null | Popis |
|---|---|---|---|
| protein | INT | True | Kolik bílkovin obsahuje 100 g jídla |
| unit_to_grams | DOUBLE | True | Převod jednotky, ve které je jídlo konzumováno, na gramy |
| unit | VARCHAR(255) | True | Jednotka, ve které je jídlo konzumováno (kus, porce, gram) |
| name | VARCHAR(255) | True | Název suroviny/jídla |
| carbohydrates | INT | True | Kolik sacharidů obsahuje 100 g jídla |
| fat | INT | True | Kolik tuků obsahuje 100 g jídla |
| id | INT | True | |

### 3.1.3 FoodMicronutrient *«table»*

Tabulka vzniklá dekompozicí M:N vazby "Jídlo" obsahuje "Mikroživina"

| Název atributu | Datový typ | Not null | Popis |
|---|---|---|---|
| micronutrient_id | INT | True | |
| amount | INT | True | Množství mikroživiny (v gramech) ve 100 gramech jídla |
| food_id | INT | True | |

### 3.1.4 Goal *«table»*

Dlouhodobý cíl, který si zákazník určí při registraci a může si ho kdykoliv změnit v nastavení aplikace.

| Název atributu | Datový typ | Not null | Popis |
|---|---|---|---|
| name | VARCHAR(255) | True | Meno cíle. |
| coefficient | DOUBLE | True | Koeficient, kterým se přenásobí vypočítaný cílený kalorický příjem uživatele na základě jeho hmotnosti, výšky a věku. |
| id | INT | True | |

### 3.1.5 Meal *«table»*

Konkrétní chod (snídaně, večeře...), při kterém zákazník jídlo konzumuje.

| Název atributu | Datový typ | Not null | Popis |
|---|---|---|---|
| time_of_day | TIME | True | Čas, ve kterém se daný chod konzumuje (např. snídaně - 9:00) |
| name | VARCHAR(255) | True | Název chodu (snídaně, večeře...) |
| id | INT | True | |

### 3.1.6 Micronutrient *«table»*

Mikroživiny, které jídlo obsahuje (vitamín C, E255, vláknina...)

| Název atributu | Datový typ | Not null | Popis |
|---|---|---|---|
| name | VARCHAR(255) | True | Názov mikroživiny |
| id | INT | True | |

### 3.1.7 Photography *«table»*

Konkrétna fotografia, ktorú zákazník môže pridať k receptu.

| Název atributu | Datový typ | Not null | Popis |
|---|---|---|---|
| id | INT | True | |
| file | VARCHAR(255) | True | Cesta k souboru, kde je uložená fotografie. |
| recipe_id | INT | True | |

### 3.1.8 Recipe *«table»*

Recept, ktorý môže zákazník vytvoriť.

| Název atributu | Datový typ | Not null | Popis |
|---|---|---|---|
| name | VARCHAR(255) | True | Názov receptu. |
| description | TEXT | True | Detailnější komentář popisující přípravu jídla. |
| recipeFood_id | INT | True | |
| user_id | INT | True | |

### 3.1.9 RecipeFood *«table»*

Vazba popisující vztah "Recept" je tvořen "Jídlem"

| Název atributu | Datový typ | Not null | Popis |
|---|---|---|---|
| food_id | INT | True | |
| amount | INT | True | Kolik jídla je obsaženo v receptu (1[kus] jogurt, 200 [ml] čaje...) |
| recipe_id | INT | True | |

### 3.1.10 ShorttermGoal *«table»*

Konkrétny krátkodobý cieľ, ktorý systém navrhne na základe informácií o zákazníkovi a ktorý si podľa potrieb zákazník upraví (nejíst cukry, chodit běhat...)

| Název atributu | Datový typ | Not null | Popis |
|---|---|---|---|
| maximum_intake | INT | False | Maximální příjem (v gramech) jídla / makronutrientu, kterého se cíl týká |
| minimum_output | INT | False | Minimální počet, kolikrát je třeba vykonat aktivitu, které se cíl týká |
| id | INT | True | |
| activity_id | INT | False | Aktivita, ke které se krátkodobý cíl vztahuje (nemusí být určena) |
| food_id | INT | False | Jídlo, ke kterému se krátkodobý cíl vztahuje (nemusí být určeno) |
| user_id | INT | True | Zákazník, který si cíl stanovil. |

### 3.1.11 User *«table»*

Osoba, ktorá navštevuje fitness centrum a využíva služby, ktoré centrum poskytuje.

| Název atributu | Datový typ | Not null | Popis |
|---|---|---|---|
| birth | DATETIME(6) | False | Dátum narodenia zákazníka. |
| email | VARCHAR(255) | True | E-mail, který zákazník sdělil na recepci, na který mu došel kód |

| | | | |
|---|---|---|---|
| weight | INT | False | Váha zákazníka. |
| code | VARCHAR(20) | True | Kód, ktorý dostane zákazník pri registrácií. |
| name | VARCHAR(255) | True | Meno zákazníka. |
| height | INT | False | Výška zákazníka |
| calories | INT | False | Cílený denní kalorický příjem, který si uživatel nastaví při registraci a může si jej změnit v nastavení |
| id | INT | True | |
| fat | INT | False | Cílený denní příjem tuků (v gramech), který si uživatel nastaví při registraci a může si jej změnit v nastavení |
| carbohydrates | INT | False | Cílený denní příjem sacharidů (v gramech), který si uživatel nastaví při registraci a může si jej změnit v nastavení |
| protein | INT | False | Cílený denní příjem bílkovin (v gramech), který si uživatel nastaví při registraci a může si jej změnit v nastavení |
| goal_id | INT | False | Dlouhodobý cíl, který si zákazník určuje při registraci (hubnutí / nabírání / udržení hmotnosti). |

### 3.1.12 UserActivity *«table»*

Tabulka vzniklá dekompozicí vazby "Zákazník vykonává aktivitu"

| Název atributu | Datový typ | Not null | Popis |
|---|---|---|---|
| activity_id | INT | True | Aktivita, která byla prováděna |
| duration | INT | True | Jak dlouho zákazník aktivitu vykonával (v jednotkách aktivity) |
| user_id | INT | True | Zákazník, který aktivitu prováděl |
| id | INT | True | |
| date | DATE | True | Den, kdy byla aktivita zaznamenána |

### 3.1.13 UserMealFood *«table»*

Tabulka vzniklá dekompozicí vazeb "Zákazník konzumuje chod" a "Chod obsahuje jídlo" = Zákazník konzumuje jídlo při daném chodu.

| Název atributu | Datový typ | Not null | Popis |
|---|---|---|---|
| meal_id | INT | True | Chod, při kterém bylo jídlo zákazníkem sněženo. |
| user_id | INT | True | Zákazník, který jídlo při daném chodu snědl. |
| food_id | INT | True | Jídlo, které bylo sněženo při daném chodu zákazníkem. |
| id | INT | True | |
| date | DATE | True | Den, ve který bylo jídlo sněženo zákazníkem při daném chodu. |
| amount | INT | True | Množství (v jednotkách jídla) jídla, které zákazník snědl při daném chodu v daný den. |