**GYMNASIUM JANA KEPLERA**

Parléřova 2/118, 169 00 Praha 6


# Název maturitní práce

Maturitní práce

Autor: Šimon Brecher

Třída: R8.A

Školní rok: 2020/2021

Předmět: Informatika

Vedoucí práce: Vedoucí práce

Praha, 2021


**GYMNASIUM JANA KEPLERA**
*Kabinet informatiky*

# ZADÁNÍ MATURITNÍ PRÁCE

| | |
|---|---|
| *Student:* | **Šimon Brecher** |
| *Třída:* | **R8.A** |
| *Školní rok:* | **2020/2021** |
| *Platnost zadání:* | **30. 9. 2021** |
| *Vedoucí práce:* | **Šimon Schierreich** |
| *Název práce:* | **Online aplikace pro evidenci výdajů rodiny** |

*Pokyny pro vypracování:*

**Vytvořte webovou aplikaci pro evidenci výdajů rodiny. Pomocí aplikace bude možné sledovat příjmy a výdaje jednotlivých členů domácnosti. Dále bude možné z výpisu z bankovního účtu roztřídit pravidelné platby a rozdělit je do kategorií, které určí uživatel. Jednorázové platby rozdělí uživatel ručně. Cílem je, aby měl uživatel přehled, za jaké kategorie utrácí kolik peněz.**

*Doporučená literatura:*

[1] EVANS, Eric. Domain-Driven Design: Tacking Complexity In the Heart of Software. Boston: Addison-Wesley Longman Publishing, 2003. ISBN 978-0-321-12521-7.

[2] MARTIN, Robert C. Design Principles and Design Patterns. www.objectmentor.com, 2000. Dostupné z: https://fi.ort.edu.uy/innovaportal/file/2032/1/design_principles.pdf.

[3] FOWLER, Martin. Patterns of enterprise application architecture. Boston: Addison-Wesley Professional, 2003. ISBN 978-0-321-12742-6.

*URL repozitáře:*

https://github.com/simonbrecher/expense_check

*vedoucí práce* *student*

*V Praze dne 30. 10. 2020*

## Prohlášení

Prohlašuji, že jsem svou práci vypracoval samostatně a použil jsem pouze prameny a literaturu uvedené v seznamu bibliografických záznamů. Nemám žádné námitky proti zpřístupňování této práce v souladu se zákonem č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších předpisů.

V Praze dne 7. dubna 2021 Šimon Brecher

## Poděkování

Děkuji Šimonu Schierreichovi za vedení mé práce, pravidelné konzultace a příhodné rady, Davidu Grudlovi za vytvoření Nette frameworku, Jakubu Vránovi za vytvoření Admineru, firmě Jetbrains za licenci na phpstorm na rok zdarma, Miloši Brecherovi za to, že mě přivedl k programování webových aplikací v php.

## Abstrakt

Abstrakt.



### Klíčová slova

finance rodiny, výdaje rodiny, za co utrácím, proč nemám peníze



## Abstract

Abstract.



### Keywords

family finances, family expenses, what do I spend money on, why do I have no money

# Obsah

# 1. Teoretická část

Můj maturitní projekt je webová aplikace pro evidenci výdajů pro rodinu. Aplikace má umožnit uživatelům mít jednodušší přehled o tom, za co utrácí peníze.

## 1.1 Cílová skupina

Aktuální verze aplikace podporuje několik rodin v databázi a několik uživatelů v rodině. Nejlépe je ale přizpůsobená pro jednoho uživatele, který rozhoduje o všem. To je především proto, že všichni uživatelé v rodině mají stejná práva. Jsou schopni upravovat a vidět pouze své doklady a platby a vidí součet výdajů pro celou rodinu. Pro lepší využití pro vyšší množství typů rodin by bylo potřeba vytvořit komplikovanější systém rolí a nejspíše i nastavení důvěry rodiny.

## 1.2 Srovnání s alternativami

# 2. Implementace

## 2.1 Logika dat aplikace

Databáze využívá cizí klíče. Když napíšu, že něco odkazuje na něco jiného, myslím tím cizí klíč.

### 2.1.1 Rodina, uživatelské účty a členové rodiny

Nový uživatel si založí svůj uživatelský účet. S ním se mu automaticky založí nová rodina. Pokud chce někdo přidat nový uživatelský účet do již existující rodiny, musí se přihlásit jako uživatel v té rodině a založit nový uživatelský účet s rolí editor, na stránce Rodina / Přidat člena rodiny.

Emailová adresa se pouze uloží do databáze, v této chvíli se k ničemu nepoužívá. Také se nijak nekontroluje, jestli existuje nebo patří uživateli. Víc uživatelů může mít stejnou emailovou adresu.

Jediné, co nemůže mít více uživatelů mimo rodinu je uživatelské heslo. Pouze to se společně s heslem používá k přihlašování.

Přihlášený uživatel může vytvořit nové členy rodiny. Ti mají roli buď spotřebitel, nebo editor. Členové s rolí spotřebitel nemají žádný účet. Uživatelé s rolí Editor mají všichni stejná práva, a proto nemůže jeden uživatel upravovat druhého. Není možné deaktivovat ani smazat uživatele aplikace.

### 2.1.2 Bankovní účty a karty

Uživatel si může vytvořit bankovní účty pouze pro Fio banku a platební karty pouze pro své bankovní účty. Více uživatelů může mít bakovní účet se stejným číslem, ale bude se počítat jako rozdílný účet. Uživatel nemůže provést import výpisu z bankovního účtu, pokud nevlastní bakovní účet, na kterým je, nebo obsahuje platební kartu, která není na tomto bankovním účtu. Uživatel může importovat výpisy z bankovního účtu pouze na své bankovní účty.

### 2.1.3 Kategorie

Uživatelé můžou vytvořit kategorie, do kterých se budou třídit výdaje. Všechny kategorie jsou společné pro celou rodinu. Využívají se v soupisu všech spotřeb, aby uživatelé věděli, za co utrácejí. Na kategorie odkazují pouze položky dokladů, díky tomu je možné dokladu přiřadit více kategorií. (Ještě trvalé příkazy odkazují na kategorie, ale to se využije k tomu, aby se roztřídili doklady.)

### 2.1.4 Podpora formátu

Jediná podporovaná měna je CZK. V databázi se všechny peníze ukládají jako decimal, ale všechny hodnoty, se kterými aplikace pracuje jsou celá čísla. Maximální počet peněz na políčko je 10 miliard korun.

Datum může mít formáty: j.n , j.n. , j.n.y , j.n.Y , d.m. (a kombinace j/d , n/m). S datem v aplikaci se pracuje jako s objektem DateTime z Nette, protože se dá jednoduše uložit do databáze, která má jiný formát data. Po získání data z formuláře se musí převést na formát j.n.Y, protože formát j.n.y by se na objekt DateTime převedl s chybou.

Jediná podporovaná banka je Fio banka. Uživatel nemůže vytvořit bankovní účet patřící k jiné bance. Banky jsou jediná data přidaná do databáze vývojářem.

### 2.1.5 Platby

Jelikož se jedná o aplikaci pro evidenci výdajů, všechny bankovní transakce se označují za platby. Ke každému dokladu se spáruje 1 nebo více plateb. V případě že útrata nebyla v rámci bankovního účtu, například v hotovosti, se vytvoří imaginární platba k dokladu.

Uživatelem vytvořené platby se do aplikace dostanou pouze pomocí importu z bankovního účtu. Není možné je přímo upravovat, nebo mazat. Nepřímo je možné upravit pouze nějaké typy plateb, a to pouze někdy.

Platby a pouze platby se označují za výdajové, nebo nevýdajové a identifikované, nebo neidentifikované.

Znaménka částek plateb jsou kladná, pokud peníze přijdou na účet, a záporná, pokud z něj odejdou.

### 2.1.6 Import výpisu z bankovního účtu

Uživatel musí pravidelně importovat výpis z bankovního účtu. Jediný formát výpisu z bankovního účtu, který je zatím možný importovat je .csv pro Fio banku.

Importovat výpis z bankovního účtu je možné pouze tehdy, kdy existuje bankovní účet a všechny platební karty, jinak se nic neimportuje. Pokud na stejném bankovním účtu existují duplicitní platby podle ID operace z banky, platba se podruhé nepřidá. Pokud import z bankovního účtu neobsahuje žádnou neduplicitní platbu, nebo je celý jeho časový interval obsažený v ostatních importech, tak se považuje celý import za duplicitní a neimportuje se. Je ale možné importovat výpis z bankovního účtu s pouze duplicitními platbami, pokud se liší v časovém intervalu.

Není možné importovat výpis z bankovního účtu, pokud okolnosti naznačují jinému formátu, nebo pokud obsahuje nepodporovanou měnu. Jediný rozdíl od očekávaného formátu, který nepřekazí import výpisu z bankovního účtu je, pokud se typ platby liší od všech rozpoznávaných, poté se typ platby určí na null.

Fio banka už jednou změnila formát výpisu z bankovního účtu tento rok - změna separátoru z čárky na středník. Doufám, že se nezmění ve velmi blízké době znovu.

Za žádných okolností neupravujte .csv soubor, především v LibreOffice.

### 2.1.7 Doklady a položky dokladů

Uživatel přídává do aplikace doklady (faktura, účtenka) pomocí formuláře pokaždé, když něco zaplatí. Ty se poté poté párují s platbami. Díky tomu má uživatel jistotu, že zaplatil všechno, co měl, a že mu nikdo z bankovního účtu nebere žádné peníze (například pomocí platební karty).

Každý doklad obsahuje vždy 1 hlavičku a 1, nebo více položek. Automaticky vytvořené doklady obsahují vždy 1 položku. Spárované platby, které odkazují na doklad, vždy odkazují na hlavičku. Jediný důvod, proč existují položky dokladů je, aby se cena dokladu mohla rozdělit do několika kategorií, nebo na několik spotřebitelů. Proto ceny jsou pouze na položkách dokladů a pouze položky dokladů odkazují na kategorie a spotřebitele.

Ve formuláři na přidání a upravení dokladů je celková cena. Ta se přepočítá na cenu hlavní položky. Hlavní položka je vždy první položka. Cena a kategorie dokladu, která se při zobrazování vykreslí, je vždy pouze celková cena všech položek a kategorie hlavní položky. Ceny, kategorie a spotřebitelé jednotlivých položek se používají při vypsání celkové spotřeby.

Znaménka cen dokladů jsou kladná, tedy opačná, než znaménka plateb. (Výjimkou je doklad typu PAIDBY_ATM, když jde o vklad hotovosti v bankomatu, nebo u přepážky. V tomto případě je znaménko záporné, protože znaménko platby je kladné – peníze totiž jdou na účet. Ale ve výpisu všech dokladů se znaménko dokladu uvádí kladně.)

### 2.1.8 Typy plateb

Typ plateb se používá pouze k automatickému párování, kde se musí schodovat, z prostorových důvodů uživatel nevidí typ při párování. Pokud uživatel manuálně spáruje dva rozdílné typy platby a dokladu, tak má přednost typ dokladu.

Typ platby je jediné, co se dá na platbě změnit, a to pouze, pokud uživatel manuálně spáruje platbu typu PAIDBY_CARD s dokladem typu PAIDBY_ATM, nebo platbu typu null.

PAIDBY_CASH

Platba odkazující na doklad v hotovosti. Vždy se vytvoří automaticky, protože neautomaticky vytvořené platby se vytváří pouze importem výpisu z bankovního účtu. Platba tohoto typu odkazuje na pokladnu uživatele, ne bankovní účet, na rozdíl od všech dalších typů.

PAIDBY_CARD

Platba kartou, nebo vklad, nebo výběr v hotovosti. (Ve výpisu z bankovního účtu Fio banky se neodlišuje.) Pokud se páruje s dokladem typu PAIDBY$_A$TM, taksepřetypuje.Dálejízměnittypužnejde.

PAIDBY_BANK

Platba bankovním převodem, nebo inkaso.

PAIDBY_ATM

Výběr, nebo vklad z bankomatu, nebo na přepážce. Jediný typ, který po párování odkazuje na doklad a je nevýdajový.

PAIDBY_FEE

Bankovní poplatek. Při importu výpisu z bankovního účtu se spáruje s automaticky vytvořeným dokladem. Tento doklad a platba nejdou nijak upravit. Doklad není pro uživatele viditelný, pouze platba. Platba tohoto typu je výdajová, proto se počítá do celkového součtu výdajů, ale nepočítá se do kategorií.

null

Tato kategorie vznikne, když při importu výpisu z bankovního účtu není možné rozhodnout, jaký typ má platba. Tato platba se automaticky nespáruje, ale dá se spárovat pomocí bankovního výpisu.

### 2.1.9 Typy dokladů

Když uživatel přidá doklad, tak musí určit jeho typ.

PAIDBY_CASH

Doklad k platbě v hotovosti. Automaticky se k němu vytvoří 1 platba v hotovosti, která se s ním spáruje. Pokud se upraví, tak se platba smaže a vytvoří znovu.

PAIDBY_CARD, PAIDBY_BANK

PAIDBY_ATM

Výběr, nebo vklad hotovosti v bankomatu, nebo na přepážce. Není možné k němu určit kategorii, spotřebitele, ani více než 1 položku.

PAIDBY_FEE

Pouze automaticky vytvořený doklad k platbě PAIDBY_FEE. Nejde zobrazit, ani upravit uživatelem.

null

Dá se vytvořit pouze automaticky při trvalém příkazu na platbu typu null.

### 2.1.10 Párování

Pro ověření správnosti všech plateb na výpisu z bankovního účtu je nutné je spárovat s doklady. Platby odkazují na doklady. Každá platba odkazuje na 1 hlavičku dokladu, nebo na null. Platba se považuje za identifikovanou, pokud má v sloupci is_identified 1. Na 1 hlavičku dokladu může odkazovat libovolný počet plateb. Doklad se považuje za identifikovaný, pouze pokud na něj odkazuje jiný počet identifikovaných plateb než 0. Není možné párovat už spárovaný doklad, nebo platbu.

Několik plateb může odkazovat na jeden doklad, ale ne obráceně. Automaticky vytvořený doklad má vždy 1 položku a automaticky vytvořené platba se vytvoří vždy 1 na doklad.

Pokud uživatel upraví doklad, tak se automaticky zruší párování. Pokud je doklad typu PAIDBY_CASH, tak se jeho automaticky vytvořená platba smaže a vytvoří se nová. Pokud chce uživatel

zrušit spárování automaticky spárované platby, tak může cenu dokladu změnit o 2, poté se platba s dokladem nespáruje automaticky. Není možné žádným způsobem upravit doklad, který byl automaticky vytvořený trvalým příkazem.

Spárovat několik plateb na 1 doklad je možné tak, že uživatel klikne na tlačítko vybrat pro několik plateb a poté na tlačítko párovat.

Doklady v hotovosti se párují automaticky s automaticky vytvořenými platbami, ale aby se platba označila za identifikovanou, tak musí uživatel rozhodnout, jestli je výdajová.

### 2.1.11 Automatické párování

Při změně databáze (ale ne při mazání dokladu) se pokusí aplikace automaticky spárovat všechny nespárované platby a doklady. (stejné je to u trvalých příkazů)

Automatické párování se provede pouze pokud se cena platby a celková cena dokladu (po obrácení znamének) liší o maximálně 1 korunu. Proto je možné trvale zrušit párování automaticky spárovaného dokladu, pokud jeho cenu uživatel upraví o 2 koruny. Pro automatické spárování se také doklad a platba musí schodovat ve dni platby/vystavení a v typu.

### 2.1.12 Trvalé příkazy

Některé platby se pravidelně opakují, a proto by bylo zbytečné je manuálně párovat. Uživatel může vytvořit trvalý příkaz na svůj bankovní účet. Trvalý příkaz automatický třídí platby, případně jim automaticky vytvoří doklad. Trvalý příkaz třídí pouze platby převodem, rozpozná je podle variabilního symbolu a určí jim kategorii. Existují výdajové a nevýdajové trvalé příkazy.

Výdajový trvalý příkaz ke každé platbě automaticky vytvoří doklad s 1 položkou, která bude odkazovat na kategorii trvalého příkazu. Trvalý příkaz nikdy nepracuje s doklady přidanými uživatelem. Výdajové trvalé příkazy vždy musí obsahovat kategorii, ale nemůžou obsahovat spotřebitele.

Nevýdajový trvalý příkaz na rozdíl od výdajového nevytváří automaticky doklad, protože nevýdajové platby (kromě výběru / vkladu a plateb v hotovosti, které se zase nepárují trvalým příkazem) neodkazují na doklad. Také nevýdajový trvalý příkaz neodkazuje na kategorii, protože na kategorie odkazují pouze položky dokladů, které se k platbě nevytvoří.

### 2.1.13 Pokladna uživatele

Uživatel by měl alespoň jednou zapsat, kolik peněz v hotovosti má. Pokladna uživatele je tabulka v databázi, která se automaticky vytvoří, když se vytvoří uživatel.

Aplikace spočítá, kolik uživatel utratil v hotovosti a kolik hotovosti vybral a vložil. Z toho spočítá, kolik by měl uživatel mít peněz. Pokud má uživatel jiné množství peněz, tak to nejspíše znamená, že něco nezapsal.

### 2.1.14 Zobrazení dat a výpis součtu výdajů

Uživatel se může podívat na seznam všech plateb a dokladů, a to pouze pro sebe. Platby vidí všechny pro každý svůj bankovní účet a své platby v hotovosti. Počítají se i automaticky vytvořené platby. U dokladů vidí všechny, kromě automaticky vytvořených dokladů typu PAIDBY_FEE.

Dále se může uživatel podívat na součet všech výdajů pro celou rodinu. Může vybrat časový úsek a rozdělit si platby podle 4 kategorií: po měsíci, po roku, podle kategorie, podle spotřebitele. V tomto případě se vyberou pouze platby, které jsou výdajové a patří časově do úseku, a od nich se přes cizí klíče přejde ke všem hlavičkám dokladů, až položkám. Z položek se získají ceny, kategorie a spotřebitelé.

V seznamu kategorií jsou uvedeny i součty cen všech položek k jednotlivým kategoriím. Tam jsou uvedeny všechny položky, včetně neidentifikovaných a nevýdajových.

## 2.2 Testování

Protože používání aplikace vyžaduje výpisy z bankovního účtu Fio banky, vytvořil jsem několik testovacích výpisů z bankovního účtu s doporučeným postupem testování. Doporučený postup testování neobsahuje platby v hotovosti a optimalizaci na testování součtu výdajů, protože to se netýká importu výpisu z bankovního účtu. Doklady a platby v hotovosti je nutné označit za výdajové, aby se objevily v součtu výdajů.

Výpis z bankovního účtu ba_import_111.csv obsahuje všechny typy plateb. Ostatní obsahují pouze jednu platbu, kvůli testování zabezpečení uživatelských dat.

Čísla ve formulářích se nemají rozdělovat mezerami. Zde někde jsou, ale to kvůli větší přehlednosti.

### 2.2.1 Doporučený postup testování

> Uživatel / Založit uživatelský účet s novou rodinou

Karel ; Jouda ; karel ; karel.jouda@gmail.com ; heslo

> Bankovní účty / Přidat bankovní účet

2010 ; 11110001 ; Aktivní

2010 ; 11110002 ; Aktivní

> Bankovní účty / Přidat bankovní účet

11110001 ; 0123 ; Karel 1 ; Aktivní

11110001 ; 1101 ; Karel 2 ; Aktivní

11110002 ; 1102 ; Karel 3 ; Aktivní

> Rodina / Přidat člena rodiny

Dítě ; Jouda ; Spotřebitel ; Aktivní

Bára ; Joudová ; Editor ; Aktivní ; bara ; bara.joudova@gmail.com ; heslo

> Kategorie / Přidat kategorii

Plyn a elektřina ; ; Aktivní

Potraviny ; ; Aktivní

Mlsy ; ; Aktivní

Dovolená ; ; Aktivní

Auto ; ; Aktivní

> Platby / Přidat příkaz

11110001 ; 111 111 ; Aktivní ; Je výdaj ; Plyn a elektřina ; Plyn – příkaz

11110001 ; 222 222 ; Aktivní ; Není výdaj ; Peníze babičce – příkaz

> Platby / Importovat výpis z bankovního účtu

ba_import_111.csv

ba_import_112.csv

ba_import_113.csv

ba_import_121.csv

> Doklady / Přidat doklad

600 ; Dovolená v hotelu ; Dovolená ; ; 1.1 ; Převodem ;

100 000 ; Auto ; ; 1.1 ; Převodem ;

600 ; Elektřina (ne příkaz) ; Plyn a elektřina ; ; 4.1 ; Převodem ;

1300 ; Kaufland ; Potraviny ; ; 16.1 ; Kartou ; **0123

1000 ; Lidl ; Potraviny ; ; 10.1 ; Kartou ; **0123

+ Přidat položku: 500 ; Mlsy ; Mlsy ; Karel

(Nyní budou doklady typu PAIDBY_ATM, které mají méně vstupů formuláře. Doporučuji nejprve nastavit typ.)

-800 ; Bankomat vklad ; 8.1 ; Výběr / vklad hotovosti

900 ; Bankomat výběr ; 9.1 ; Výběr / vklad hotovosti

-1100 ; Přepážka vklad ; 10.1 ; Výběr / vklad hotovosti

1200 ; Přepážka výběr ; 14.1 ; Výběr / vklad hotovosti

> Platby / Párovat platby s doklady

Kaufland – Kaufland (Typ platby nebyl rozpoznán, proto je null a nespáruje se automaticky.)

Hotel + Dovolená – Dovolená v hotelu (Více plateb se spáruje s 1 dokladem. Neprovádí se automaticky.)

Tři kladné platby označte za nevýdajové, měli by mít popisek: Příjem A, Příjem B, Příjem C.

Všechny 4 platby na vklad / výběr by měly být spárované automaticky.

Mělo by zbýt 5 neidentifikovaných plateb, 2 z nich jsou neznámé a 3 z jiných bankovních výpisů.

> Odhlásit se

> Přihlásit se

bara ; heslo

> Bankovní účty / Přidat bankovní účet

2010 ; 22220001 ; Aktivní

> Bankovní účty / Přidat platební kartu

22220001 ; 2201 ; Bára 1 ; Aktivní

> Platby / Importovat výpis z bankovního účtu

ba_import_211.csv

> Odhlásit se

> Uživatel / Založit uživatelský účet s novou rodinou

Hacker ; Hacker ; hacker ; hacker@gmail.com ; heslo

> Bankovní účty / Přidat bankovní účet

2010 ; 55550001 ; Aktivní

> Bankovní účty / Přidat platební kartu

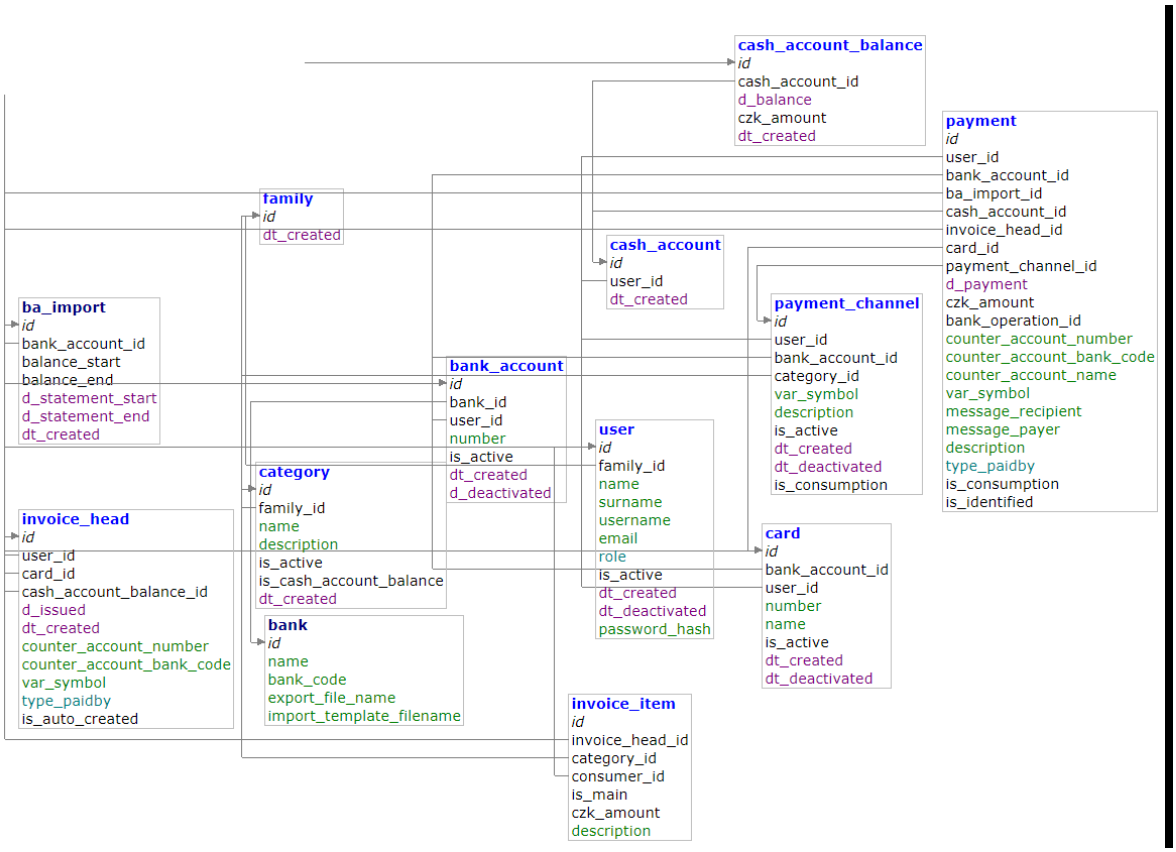
55550001 ; 5501 ; Hacker 1 ; Aktivní

> Platby / Importovat výpis z bankovního účtu

ba_import_511.csv

### 2.2.2 Schéma databáze

[t]0.49



# 3. Instalace

Stáhněte a nainstalute si MySQL Community Server 8.0.17

https://dev.mysql.com/downloads/mysql/

Apache 2.4.39 win64 VS16, poslední balíček C++ vc_redist.x64.exe for Visual Studio 2015-2019

https://www.apachelounge.com/download/

PHP 8.0.3: php-8.0.2-Win32-vs16-x64 (VS16 x64 Thread Safe)

https://windows.php.net/downloadphp-8.0

Adminer poslední verze pro php 8.

https://www.adminer.org/cs/

Stáhnete si Nette Sandbox a nainstalujte ho podle návodu:

https://github.com/nette/sandbox

Stáhněte si můj repozitář a přidejte všechny složky do souborů s odpovídajícími názvy.

Vytvořte si v Admineru databázi s porovnáváním utf8mb4_czech_ci a importujte do ní soubor documentation/myfin.sql

Vytvořte v Admineru uživatele, který bude moct upravovat databázi. Dejte mu práva: insert, update, delete a select.

Zadejte údaje pro uživatele do souboru app/config/anonymous_local.neon a přejmenujte ho na local.neon.

Na localhostu vyhledejte url adresu pro soubor www/index.php.

# 4. Závěr

Moje nejhorší chyba byla vybrat si formát výpisu z bankovního účtu, který jsem si vybral.

## 4.1 Budoucí vývoj

Kdybych tuto aplikaci někdy chtěl používat, tak bych ji napsal celou znovu, protože je spoustu věcí, které bych nyní udělal lépe.

Například vytvořit třídu pro dynamický formulář, který by se zeptal na potvrzení po zmáčknutí tlačítka, abych uživatel nemusel být přesměrovaný na jinou stránku.

Určitě by bylo vhodné umožnit uživateli nějakým způsobem upravovat, nebo mazat importy výpisu z bankovního účtu, na druhou stranu zrovna platby, které vyšli z bankovního účtu nejsou něco, co by mohl uživatel špatně přidat, aniž by přepisoval výpis z bankovního účtu.

Je potřeba vytvořit nastavení pro důvěru rodiny, po rozsáhlé analýze.

Vzhledem k časovému omezení povahy maturitní práce nebylo možné dlouhodobě testovat aplikaci na skutečných datech.

Pro lepší přehlednost by bylo výhodné upravit grafické rozhraní.

Aplikace má přehled o výdajích, ale ne o příjmech. Dále by se hodilo podporovat komplikovanější způsoby přesunu peněz.

# Seznam použité literatury


[Ein05] Albert Einstein. "Zur Elektrodynamik bewegter Körper". In: *Annalen der Physik* 322.10 (1905), s. 891–921. DOI: http://dx.doi.org/10.1002/andp.19053221004.

[GMS93] Michel Goossens, Frank Mittelbach a Alexander Samarin. *The LaTeX Companion*. Reading, Massachusetts: Addison-Wesley, 1993.

[Knu] Donald Knuth. *Knuth: Computers and Typesetting*. URL: http://www-cs-faculty.stanford.edu/%5C~%7B%7Duno/abcde.html.

# Seznam obrázků

# Seznam tabulek