1. Popis aplikácie

Expense tracker je webová aplikácia na sledovanie vývoja hodnoty osobného majetku. Je určená pre bežného človeka ktorý sa chce naučiť lepšie hospodáriť so svojimi financiami, vyhodnocovať výdaje a sledovať prijmi. Bežný človek funguje tak že mu dôjde výplata a následne do ďalšej výplaty bezhlavo míňa peniaze bez určitého plánu, kedže uchovávať každú transakciu v hlave je takmer nemožné. Potom sa ľudia spytujú samy seba že kam tie peniaze zmizli. Niektoré banky na to majú spravené rozhrania kde sa dajú transakcie zaraďovať do kategórii, ale sú dostupné iba pre klientov banky. Ja som to napriklad skúšal vyriešiť zapisovaním do excelu a predpokladám že aj viacero iných ľudí. Problém je ale že Excel ponúka priveľa možností a nemá priateľské rozhranie pre človeka ktorý s ním nerobí dennodenne. Inšpiroval som sa aj myšlienkou aplikácií ktoré slúžia na trackovanie kalórií alebo od aplikácie na zapisovanie cvikov spolu s hmotnosťami a opakovaniami.

Používateľ sa prihlási na stránke s autentifikáciou pomocou google účtu. Tento spôsob zjednodušuje registráciu tým že si netreba vymýšľať prihlasovacie meno a heslo. Taktiež sa takto vyhnem tomu aby som musel uchovávať používateľové prihlasovacie údaje. Prihlásenie pretrvá aj po zavretí stránky a opätovnom otvorení. Prihlásenému užívateľovi sa budu ukladať všetky vykonané zmeny. Novému užívateľovi sa automaticky nastaví meno a email z Google účtu.

Tieto údaje si bude môcť zmeniť a môže si nastaviť aj hlavnú menu v ktorej sa bude jeho majetok počítať, pričom predvolené bude euro. Bude možné si vybrať preferovaný formát dátumu a času. V rozhraní stránky si používateľ pridá jednotlive zdroje financí ktoré vlastní ako sú kreditné karty, hotovosť, kryptopenazenky, bankové účty. Pri vytváraní budu preddefinované typy napriklad pri výbere bankového účtu sa bude dať zapísať IBAN. názov banky, pri kreditnej karte číslo karty dátum expirácie, pri brokerových stránkach URL stránky. Všetky budú mať spoločné pole aktuálneho stavu, menu v akej má financie držané, poznámku, voliteľný názov, výber ikony. Bude možné pre jeden zdroj pridať viacero stavov a mien v prípade že drží kombináciu rôznych mien.

Na jednej podstránke bude možné vytvárať kategórie transakcií ako napríklad potraviny, výlet, živnosť a zábava. Polia na vyplnenie budú názov a ikona. Voliteľné bude možné nastaviť budget na mesačný alebo ročný interval. Po prekročení množstva výdajov pre kategóriu s budgetom sa užívateľovi zobrazí upozornenie.

Následne bude možné vytvárať, vymazať a editovať záznamy transakcií prijmov, výdajov, nákupov, predajov, dlžôb a transferov z jedneho zdroja na druhý. V prípade prijmov a výdajov bude možné vybrať z akeho zdroja boli peniaze odpočítané, vybrať kategóriu, deň a čas, poznámku, názov a pripojiť súbor. Nákup je typ transakcie ktorý sa bude využívať hlavne pri investíciách kde síce odídu peniaze ale z transakcie dostanem naspäť akciu alebo krypto. Bude možné vybrať zdroj z ktorého ubudli peniaze, poplatky, deň a čas. Následne bude možné pridať do zoznamu položky ktoré používateľ obdržal. Bude možné vybrať z možností akcie, mena, kryptomeny a NFT. Používateľ zapíše množstvo, názov, menu, poznámku a cenu jedného kusu. V pripade nakupu akcie za inú menu sa dá nastaviť konverzný kurz. Predaj bude fungovať rovnako pričom sa bude dať vyplniť cena predaja a cieľový účet kam boli peniaze odoslané. Transfer bude mať na výber, zdroj, cieľ, poplatok.

Pri investíciách sa pri akciách a kryptomenách budú načítavať udaje z verejnej API. Ak hodnotu nebude možné zistiť z API, tak si používateľ môže nastaviť manuálne aktualnu hodnotu alebo sa bude brať ta hodnota akú mala pri nákupe.

V transakcii prijmu a vydaju bude možné nastaviť rekurenciu na báze dňa, týždňa, mesiaca, roka. Transakcie bude možné filtrovať podľa názvu, kategórie, typu transakcie, veľkosti transakcie a časového rozmedzia. V adresári bude možné vytvárať entity dlžníkov a prijímateľov. Tu sa bude dať zapísať meno a číslo účtu. Entita z adresára môže byť pridaná do transakcie.

Pri dlžobe sa bude dať vybrať entita z adresára, množstvo popis a výber či je používateľ dlžník alebo naopak. Po splatení dlžoby tuto zmenu upraví v transakcii a dlžoba bude vybavená.

Ďalšia funkcionalita je nastavenie odosielania e mailových reportov na mesačnej, ročnej a týždennej báze. Budú obsahovať zhrnutie transakcií za toto obdobie.

Ak používateľ má účet v banke VUB tak si môže v ich IB exportovať transakcie a následne naimportovať do aplikácie. Dodatočne môže priraďovať kategórie a nastaviť ďalšie polia.

Pre zobrazenie štatistiky bude spravená ďalšia podstránka kde bude predvolené časové obdobie od prvej transakcie až do prítomnosti. V grafe bude zobrazený vývoj hodnoty majetku v tomto časovom období. V zhrnutí bude zobrazené akú hodnotu mali jednotlivé typy transakcii.

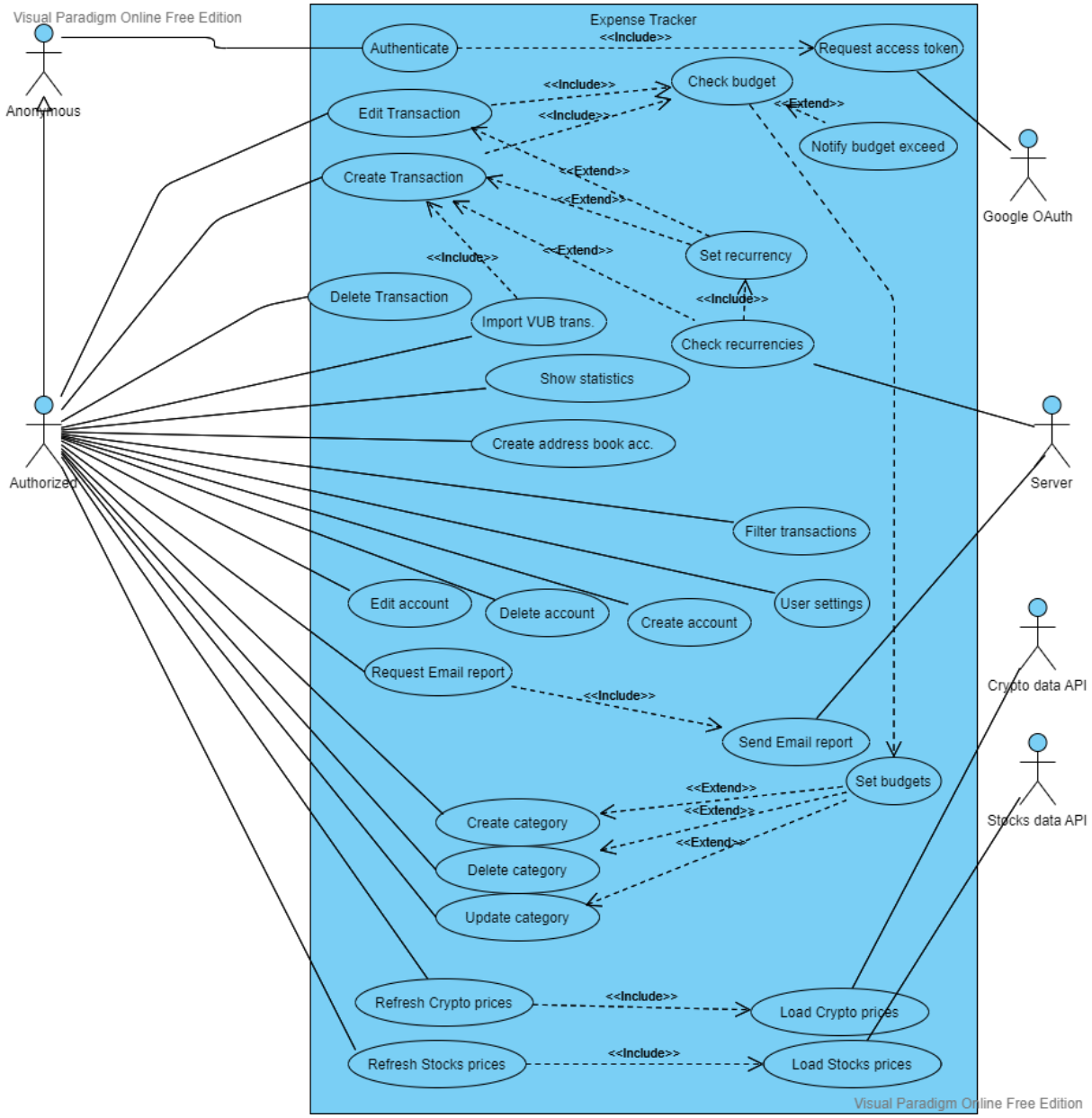
2. Typy používateľov

Používateľ ktorý nie je prihlásený cez Google autentifikáciu nebude môcť využívať stránku. Po prihlásení sú všetci používatelia rovnocenný čiže existuje iba jedna rola *Authorized user*. Obsah ktorý tento používateľ vytvorí môže vidieť a meniť iba on. Dostupné funcionality sú popísane v sekcií 1.

3. Ostatné systémy

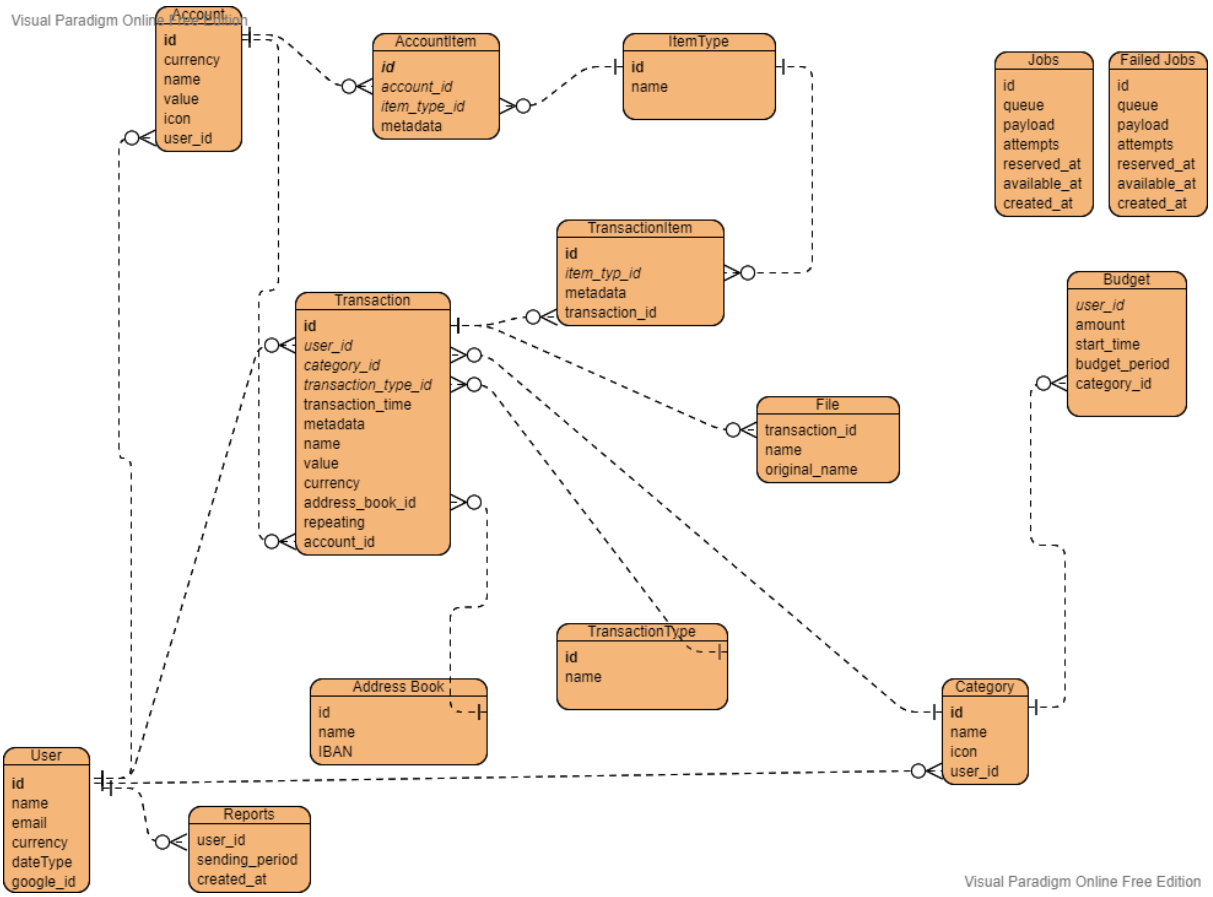
Pri zisťovaní hodnoty akcií sa budú brať údaje z REST API marketstack.com. Hodnoty kryptomien sa budú získavať z coingecko.com. V prípade že by údaje boli nedostatočné tak by som skúsil iné stránky ktoré poskytujú API rozhranie. Používateľ bude mať call to action tlačítko na prepočítanie hodnoty akcií. Po kliknutí sa spustí asynchrónny job v súlade limitami API a aktualizuje aktualne ceny akcii a kryptomien pre pouzivatela. Pri prihlaseni budem používať Google OAuth.

4. Use case diagram



5. Databáza

Databázu bude používať MySQL 8 ktorá podporuje JSON data type. Takto nebudem musieť používať relácie tam kde to nebude dávať zmysel a iba by sa skomplikovala štruktúra databázy. Napriklad ked že bude existovať viacero druhov transakcii s rôznymi poliami, musel by som vytvárať buď zvlášť tabuľku pre každý typ transakcie alebo zaznam transakcie joinovat na ďalšiu tabuľku kde by boli dodatočné polia. Vyhnem sa tomu tak že tieto odlišné polia budú serializované a uložené v JSON poli. MySQL má taktiež dobrú podporu ORM Eloquent ktoré budem používať na backende.



6. Tabuľky databázy

id stĺpec v tabuľkách je autoincrement ktorý bude aj PK

6.1 User

Obsahuje záznamy používateľov

- name - meno používateľa
- email - email používateľa
- currency - hlavná mena v ktorej sa bude zobrazovať celkova hodnota majetku
- dateType - vybraný formát dátumu a času
- google_id - id z google účtu

Tabuľka ma One-to-Many vzťah s tabuľkou **Account**, **Category**, **Transaction**

6.2 Transaction

Obsahuje transakcie používateľov

- FK *user_id* - ID používateľa
- FK *category_id* - ID príslušnej kategórie
- FK *transaction_type_id* - ID typu transakcie

- *transaction_time* - Timestamp vykonania transakcie
- *metadata* - JSON pole v ktorom budú uložené hodnoty podľa príslušného transaction typu
- name - Názov transakcie
- value - Množstvo peňazí použitých v transakcií
- currency - Mena použítá v transakcií
- FK adress_book_id - ID záznamu z adresára použitého v transakcií
- repeating - Názov spôsobu opakovania transakcie
- FK account_id - ID účtu priradeného k transakcií

Tabuľka má Many-to-One vzťah s tabuľkou **Address Book**, **Transaction Type**, **Category**

6.3 Account

Obsahuje účty používateľa ako napriklad peňaženka, kreditná karta, bankový účet.

- currency - Hlavná mena účtu
- name - Názov účtu
- value - Predpočítana hodnota účtu
- icon - Vybraná ikona účtu
- FK user_id - ID používateľa

Tabuľka ma One-to-Many vzťah s tabuľkou **Account Item**, a **Transaction**.

6.4 Category

Obsahuje vytvorené kategórie používateľom.

- name - Názov kategórie
- icon - Ikona priradená ku kategórií
- FK user_id - ID používateľa ktorý kategóriu vytvoril

Tabuľka ma One-to-Many vzťah s tabuľkou **Budget**

6.5 Budget

Tabuľka obsahuje budgety nastavené používateľom a priradené ku kategórií.

- FK user_id - ID používateľa
- budget_period - Časový interval na ktorý sa bude budget kontrolovať pri vytvorení transakcie (týždeň, deň)
- amount - Hodnota budgetu ktorá by sa nemala presiahnuť
- start_time - Čas vytvorenia pravidla budgetu
- FK category_id - ID kategórie na ktorú sa aplikuje budget

6.6 Account Item a Transaction Item

Tabuľky obsahujú vlastnené položky účtov alebo transakcií ako sú akcie, kryptomeny.

- FK account_id - ID účtu ktorý vlastní položku
- FK transaction_id - ID transakcie ktorá vlastní položku
- FK item_type_id - ID z tabuľky ItemType, vyber definovaného typu položky
- metadata - JSON pole uchovávajúce údaje položky podľa toho aké údaje má definované typ položky

Tabuľka má Many-to-One vzťah s tabuľkou **ItemType**

6.7 Item Type

Tabuľka obsahuje definované typy položiek (akcie, kryptomeny) ktoré sú implementované na Backende.

6.8 Address Book

Adresár osôb alebo objektov. Obsahuje ID, názov a IBAN číslo. Položky z adresára je možné vyberať pri transakciách.

6.9 Transaction Type

Typy transakcií ktoré sú implementované na backende a to výdaj, prijem, prevod, nákup. Typ transakcie sa vyberá pri vytváraní záznamu transakcie. Tabuľka obsahuje iba názov.

6.11 File

Obsahuje súbory priradené k transakciam. Má polia: FK transaction_id, name - vygenerovaný názov súboru a original_name čo je pôvodný názov súboru.

6.11 Reports

Obsahuje pravidlá posielania emailových reportov. Má polia: FK PK user_id, sending_period mesacne, rocne, tyzdenne a created_at timestamp kedy bolo pravidlo vytvorené.

6.10 Jobs a Failed_Jobs

Samostatné tabuľky ktoré vytvorí Laravel pri vykonávaní Queued Job-ov prostredníctvom databázy.

- id - ID jobu
- queue - Nazov fronty
- payload - Objekt ktorý bude skonzumovaný pri vykonaní úlohy
- attempts - Počet pokusov na vykonanie úlohy
- reserved_at, available_at, created_at - Timestamp údaje

7. Technologické požiadavky

Aplikácia bude mať na backende PHP framework Laravel s knižnicou Laravel Livewire ktorá je určená na vytváranie moderných, reaktívnych, dynamických rozhraní využívaním Laravel

Blade čo je technológia template viewov. V kombinácií s Livewire budem používať Javascriptovú knižnicu Alpine.js. Na štýlovanie frontendu budem používať CSS framework Tailwind. Databáza bude použitá MySQL 8 prostredníctvom ORM Laravel Eloquent. Prehliadače budú plne podporované všetky okrem Internet Exploreru a Safari. Aplikácia je primárne určená pre desktopové zariadenia.

8. Časový plán

Týždeň 5. (8.3 - 14.3)

- Vytvorenie základnej kostry projektu a nastavenie vývojového prostredia - 2h
- Vytvorenie modelov, ciest, tried - 4h
- Vytvorenie migrácii tabuliek s príslušnými poľami a ich importovanie do databázy - 4h

Týždeň 6. (15.3 - 21.3)

- Metódy na vytváranie transakcií, účtov, adresára a ukladanie nastavení - 8 h
- Vytvorenie frontendu pre podstránky Účty, Nastavenia, Kategórie, Transakcie - 6h

Týždeň 7. (22.3 - 29.3)

- Implementácia typov položiek Akcia, Kryptomena - 4 h
- Implementácia typov transakcií Výdaj, Prijem, Prevod, Nákup, Dlžoba - 6 h
- Frontend na pridávanie položiek k účtom a transakciam - 2h
- Stránka so štatistikami - 3h
- Filtrovanie transakcií - 2 h

Týždeň 8. (30.3 - 4.4)

- Úprava a vymazanie záznamov transakcii, účtov, kategórií - 3h
- Implementácia pridávania položiek k účtom a transakciam spolu s podporou kombinácii mien - 3 h
- Nastavenie budgetov pri kategóriách a ich kontrola pri vytváraní transakcií - 3h
- Nahrávanie súborov pri vytváraní transakcie - 1h

Týždeň 9. (5.4 - 11.4)

- Prihlasovanie pomocou OAuth, zistenie fungovania, implementácia - 2h
- Ukladanie prihlásenia po ukončení stránky - 2h
- Odhlásenie - 2h
- Nastavenie preferovaného zobrazovania času a následná aplikácia na stránkach - 1h
- Opakujúce sa transakcie - 3h
- Vyladenie Frontendu - 1h

Týždeň 10. (12.4 - 18.4)

- Import excelu z VUB exportu a vytvorenie transakcií k jednotlivým položkám - 3h
- Implementácia načítania údajov o akciách z API - 3h
- Implementácia načítania údajov o kryptomenách z API - 3h
- Implementácia spustenia asynchrónneho procesu na získanie údajov po kliknutí používateľom - 1 h
- Odosielanie e-mailových reportov - 2h

Týždeň 11. (19.4 - 25.4)

- Dokončovanie - 5 hodín
- Testovanie - 6 hodín

Spolu by to malo byť 85 hodín na dokončenie všetkých zaumienených funkcionalít.