Katedra informatiky
Přírodovědecká fakulta
Univerzita Palackého v Olomouci

BAKALÁŘSKÁ PRÁCE

Multiplatformní aplikace pro správu osobních financí

2025

Vojtěch Netrh

Vedoucí práce:
doc. RNDr. Jan Konečný, Ph.D.

Studijní program: Informatika,
Specializace: Programování a vývoj software

Bibliografické údaje

Autor:	Vojtěch Netrh
Název práce:	Multiplatformní aplikace pro správu osobních financí
Typ práce:	bakalářská práce
Pracoviště:	Katedra informatiky, Přírodovědecká fakulta, Univerzita Palackého v Olomouci
Rok obhajoby:	2025
Studijní program:	Informatika, Specializace: Programování a vývoj software
Vedoucí práce:	doc. RNDr. Jan Konečný, Ph.D.
Počet stran:	37
Přílohy:	elektronická data v úložišti katedry informatiky
Jazyk práce:	český

Bibliographic info

Author:	Vojtěch Netrh
Title:	Cross-platform application for personal finance management
Thesis type:	bachelor thesis
Department:	Department of Computer Science, Faculty of Science, Palacký University Olomouc
Year of defense:	2025
Study program:	Computer Science, Specialization: Programming and Software Development
Supervisor:	doc. RNDr. Jan Konečný, Ph.D.
Page count:	37
Supplements:	electronic data in the storage of department of computer science
Thesis language:	Czech

Anotace

Finance v dnešní době hrají v životech lidí podstatnou roli. Mít v nich pořádek přináší řadu benefitů. Člověk zjistí, zda-li neutrácí někde zbytečně či jaký má přebytek. Digitální doba je ideální pro využití aplikací na jejich správu. Navržená multiplatformní aplikace Budget Buddy tyto potřeby naplňuje. Umožňuje uživateli jednoduše evidovat příjmy a výdaje na denní bázi. Poskytuje také nástroje pro následnou analýzu v podobě grafů a zajímavých číselných údajů. V neposlední řadě je data možné exportovat do CSV a JSON formátu pro další použití. Využití multiplatformního frameworku Flutter zajišťuje dostupnost aplikace pro různé operační systémy.

Synopsis

Finance plays a significant role in people's lives today. Keeping them in order brings a number of benefits. A person can determine whether they are spending unnecessarily or what their surplus is. The digital age is ideal for using applications to manage finances. The proposed cross-platform application, Budget Buddy, fulfills these needs. It allows the user to easily record income and expenses on a daily basis. It also provides tools for analysis in the form of graphs and interesting numerical data. Last but not least, the data can be exported to CSV and JSON formats for further use. The use of the cross-platform framework Flutter ensures the application's availability for various operating systems.

Klíčová slova: Flutter; Dart; multiplatformní; osobní finance

Keywords: Flutter; Dart; cross-platform; personal finance

Děkuji panu doc. RNDr. Janu Konečnému, Ph. D. za cenné rady a ochotu při tvorbě práce. Děkuji mé rodině za podporu během celého bakalářského studia.

Odevzdáním tohoto textu jeho autor místopřísežně prohlašuje, že celou práci včetně příloh vypracoval samostatně a za použití pouze zdrojů citovaných v textu práce a uvedených v seznamu literatury.

Obsah

Seznam obrázků

1 Úvod

Peníze se vyskytují všude kolem nás a chceme-li nebo ne, hrají v našich životech podstatnou roli. Z pohledu jednotlivce je užitečné mít ve financích pořádek. Můžeme tak ušetřit peníze, případně je efektivněji využívat. V neposlední řadě by měl člověk vědět, kolik peněz by potřeboval v případě výpadku příjmů.

V dnešní době již většina lidí používá internetové bankovnictví, které často poskytuje alespoň základní statistiky o nakládání s financemi. Na druhou stranu lidé mívají účty u více bank a analýzy o stavu financí přímo v bankovních aplikacích nejsou dostatečně přizpůsobitelné. Lidé proto vyhledávají alternativy v podobě aplikací třetích stran. V těchto aplikacích mají uživatelé plnou kontrolu nad tím, co si evidují a lepší podmínky pro analýzu.

1.1 Požadavky

Aplikace by měla vyplnit nedostatky již existujících řešení. Smyslem je navrhnout uživatelsky přívětivou aplikaci, která bude velmi pohodlná při denním používání a bude vyžadovat velmi málo času pro zadávání dat. Pro analýzu dat by měla poskytnout dostatečně komplexní nástroje, ideálně s vhodnou vizualizací.

- **Jednoduchost více než komplexní funkce** – snažit se implementovat dostatečný počet funkcí, avšak nezahltit uživatele příliš mnoha různými nastaveními, které mu přidají práci při volbě parametrů. Ideálně pokud je pro uživatele pohodlné zapisovat transakce již v průběhu dne, aniž by měl pocit, že u aplikace tráví moc času.

- **Mobilní telefony i počítače** – zpřístupnění aplikace na více různých platformách a zařízeních přináší dostupnost aplikace pro více uživatelů. Někdo preferuje evidenci financí na denní bázi (obvykle pomocí mobilního telefonu), někdo naopak až měsíc zpětně. Na zpracování více dat najednou je jistě počítač s větším displejem vhodnější volbou.

- **Zahraniční měny** – v dnešní době velká část lidí cestuje jak pracovně, tak i za dovolenou. Placení kartou v zahraničí bez nutnosti dopředu směnit hotovost se stalo standardem. Jelikož každá banka používá jiný směnný kurz měny, měl by jít libovolně upravit, případně automaticky synchronizovat z internetu.

- **Export dat** – uživatel by měl mít možnost exportovat data v běžně používaných formátech jako je CSV[1] nebo JSON[2]. Export považuji za důležitý z důvodu přenesení dat do jiné aplikace, pro další analýzu nebo pro zálohu dat.

[1] Comma-separated values
[2] JavaScript Object Notation

2 Přehled existujících řešení

Pro organizaci financí již existuje řada aplikací poskytujících tuto funkcionalitu. Každé řešení přistupuje k problému jiným způsobem.

Nejblíže je z hlediska uživatele poskytnutí základních nástrojů přímo v bankovní aplikaci. Z mého pohledu je to nejméně vhodné řešení hned z několika důvodů:

- obvykle má člověk účet u více bank (aplikace jedné z nich neposkytuje přehled o celkových financích),
- jen část (i když v dnešní době většinová) plateb je prováděna pomocí platební karty,
- uživatel nemusí chtít mít všechny pohyby na účtu zahrnuty do celkové analýzy.

Aplikace třetích stran v této oblasti nabízí různé možnosti. Mobilní aplikace bývají obvykle jednodušší s přívětivějším uživatelským rozhraním. Zatímco desktopové aplikace mají uživatelské rozhraní nemoderní, avšak poskytují nepřebernou škálu funkcí.

2.1 1Money

Ještě na konci roku 2024 byla aplikace 1Money dostupná na obě mobilní platformy, o pár měsíců později už aplikace funguje jen uživatelům, kteří ji měli nainstalovanou dříve. Z těchto důvodů není možné poskytnout žádný odkaz na aplikaci, i když na mém zařízení pořád funguje bez problémů. Z mnou vyzkoušených aplikací se jedná o největšího favorita pro denní používání. Aplikace mě zaujala svojí obrazovkou *Categories* (viz Obrázek 1). Středem obrazovky je prstencový graf zobrazující poměr utracených peněz podle kategorií a v jeho středu se nachází dva údaje – stav příjmů a výdajů. Zbytek obrazovky pokrývají kolečka označující kategorie, po jejichž stisknutí je uživateli umožněno přidat transakci s danou kategorií a aktuálním časovým razítkem. Umožňuje vytvořit více účtů, včetně účtů pro spoření. Z hlediska analýzy a grafů zde najdeme pouze dvě velmi omezené možnosti – již zmíněný prstencový graf kategorií a sloupcový graf znázornění transakcí v čase.

2.2 Cashew

Multiplatformní aplikace, založená na stejné technologii Flutter jako moje aplikace, se jmenuje Cashew [1]. Implementuje komponenty ze systému Material Design (viz kapitola 3.3 Material Design a Obrázek 2), ale rozvržení obrazovky si autoři uspořádali podle sebe. Uživateli umožňuje široké možnosti nastavení z hlediska vzhledu (barvy, fonty, typ ikon či formáty čísel). Umožňuje zálohování

Obrázek 1: Aplikace 1Money

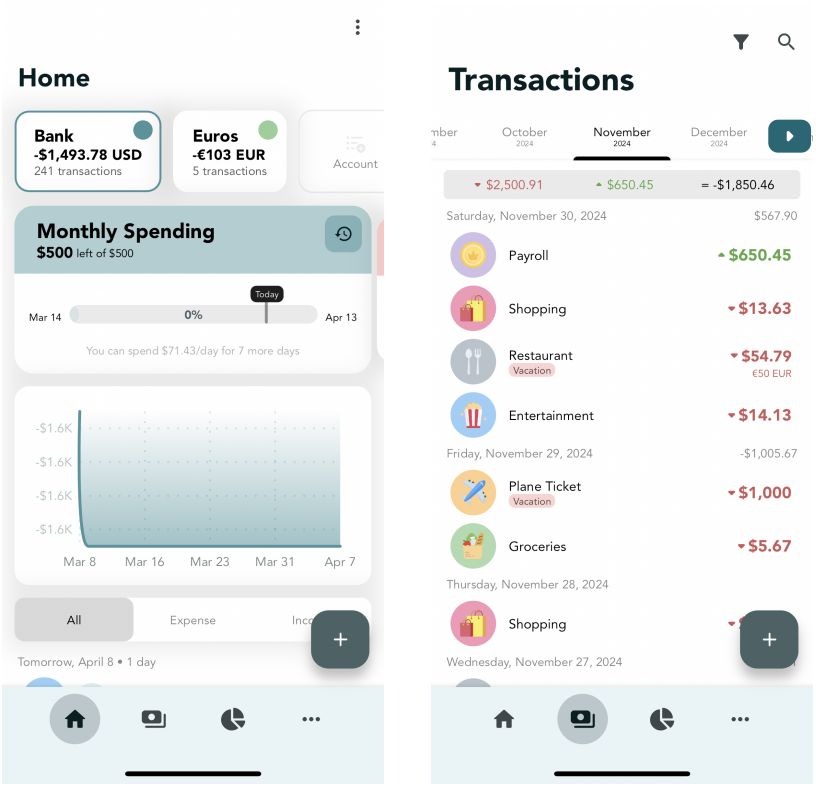
dat na Google disk i import z CSV souborů. Podporuje funkci více účtů. Nabízí i placenou PRO verzi, která přináší jen málo funkcí a slouží především jako podpora vývojářů.

2.3 Wallet

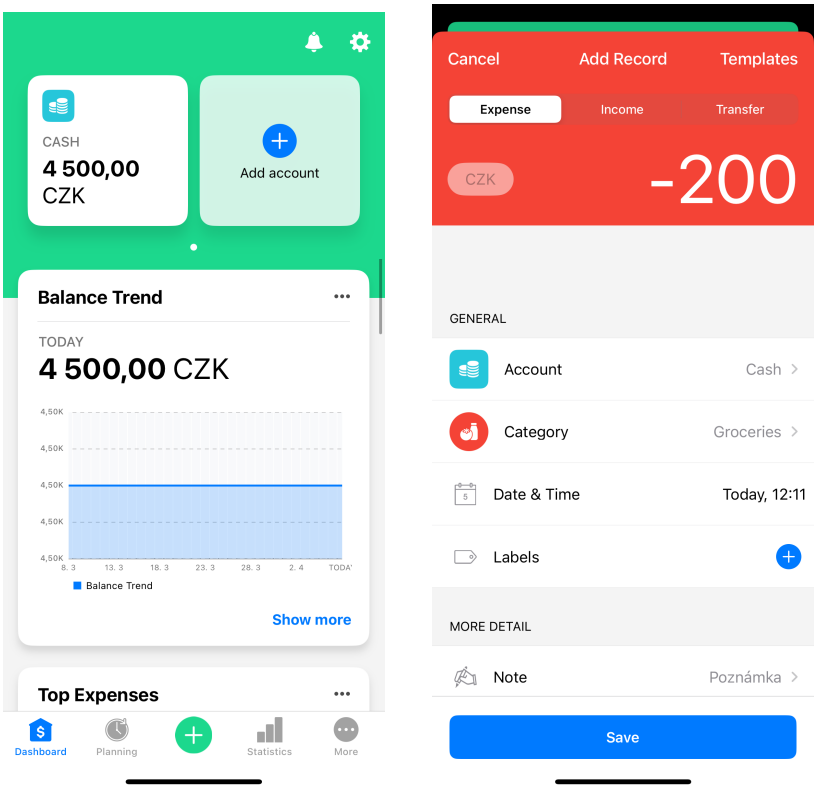
Mobilní aplikace Wallet [2] je dostupná na platformy Android i iOS. Za její stažení uživatel v první fázi nezaplatí. Nabízí všechny základní funkce, co by uživatel očekával – kategorie transakcí, podpora více měn, přidání poznámky. Z hlediska analýzy a grafového zobrazení je na výběr jen velmi málo možností v nepřehledném uspořádání (viz Obrázek 3). Ze zajímavých funkcionalit stojí za zmínku plánované transakce nebo automatická bankovní synchronizace (součást placené varianty).

2.4 Money Manager Ex

Money Manager Ex [3] je aplikace dostupná jak na mobilní telefony, tak především na desktopová zařízení. Uživatelské rozhraní je na první pohled relativně přívětivé, i když místy obsahuje širokou škálu tlačítek a možností k nastavení. Při prvním spuštění nevyžaduje žádné složité nastavení, jak je vidět na Obrázku 4. Zde se mi velmi líbí velká nabídka grafů k analýze údajů a možnost exportu



Obrázek 2: Aplikace Cashew



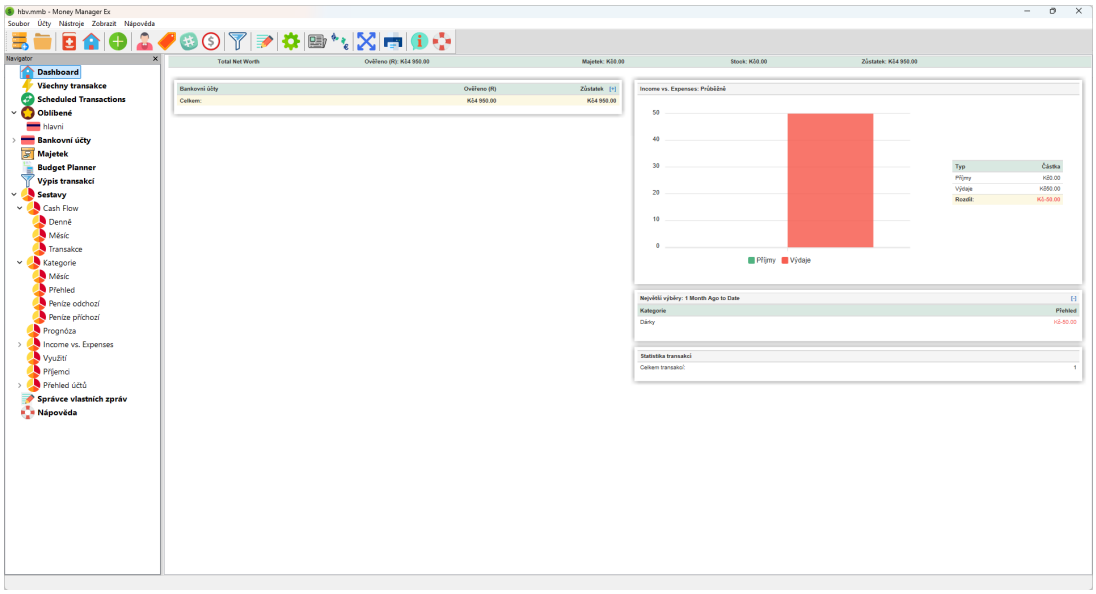
Obrázek 3: Aplikace Wallet

do PDF formátu pro většinu obrazovek.

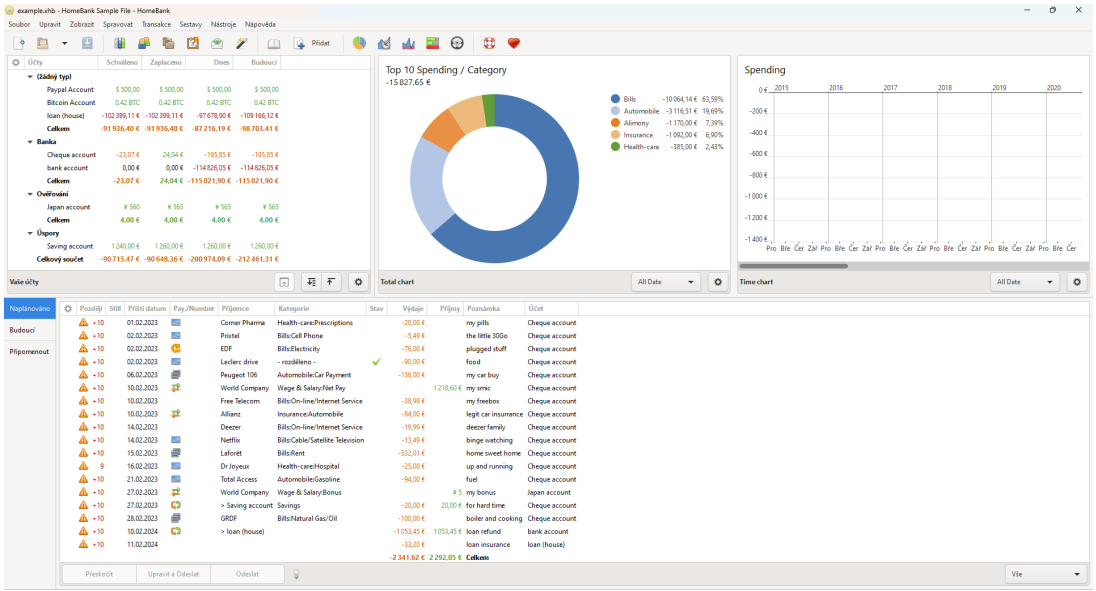
2.5 Homebank

Desktopová aplikace Homebank [4] je dostupná na Windows, Linux i macOS. Z hlediska uživatelského komfortu je oproti Money Manager Ex jednodušší na používání. Při prvotním nastavení sice člověk musí vhodně nakonfigurovat svůj účet, v dalších fázích je už používání jednoduché. Centrem aplikace je obrazovka (viz Obrázek 5) se stavy účtu, přehledem transakcí a jednoduchými grafy. Všechny ostatní funkce se pak otevírají jako nové okno. Uživatelské rozhraní působí moderním a intuitivním dojmem.

Oproti existujícím aplikacím by mnou vytvořená aplikace měla nabídnout přívětivější uživatelské rozhraní a ulehčit první použití. Rád bych přinesl i lepší vizualizaci v podobě grafů a zajímavých číselných údajů. V neposlední řadě by měla být spustitelná na libovolné platformně, tak aby uživatel nebyl limitován například při změně mobilního telefonu.



Obrázek 4: Aplikace Money Manager Ex



Obrázek 5: Aplikace Homebank

3 Použité technologie

Hlavním požadavkem byl multiplatformní vývoj. K této problematice existuje řada technologií [5], které však k tvorbě aplikace přistupují odlišným způsobem. Mezi nejpopulárnější technologie vhodné na tuto práci se řadí *React Native* a *Flutter*. Jelikož jsem za dobu, co se věnuji programování nepřilnul k webovým technologiím, na kterých primárně stojí React Native, zvolil jsem framework Flutter. Přispěl k tomu fakt, že se více blíží klasickým objektově orientovaným jazykům a jednoduše implementuje systém Material Design, který rád v uživatelském rozhraní vídám.

Aplikace je otestována a řádně funguje na mobilní platformě Android a jako desktopová Windows aplikace. Ze stejného kódu jde sestavit i aplikace na iOS, macOS a Linux. Tyto platformy jsem však netestoval a nemohu zaručit jejich bezproblémový chod.

3.1 Dart

Objektově orientovaný jazyk Dart [6] je vyvíjen společností Google. První verze byla zveřejněna 14. listopadu 2013, aktuální je verze 3. Jde o typově bezpečný jazyk, podporující třídy, se syntaxí založenou na jazyku C (viz Zdrojový kód 1). Může být kompilován do strojového kódu, jazyků JavaScript nebo WebAssembly. Taktéž je základem pro framework Flutter.

Je dodáván se širokou škálou základních knihoven, které umožňují obstarat běžný vývoj. Jako příklad uvádím knihovnu `dart:async`, která umožňuje asynchronní programování za využití tříd jako `Future` nebo knihovna `dart:io` pro práci se souborovým systémem či HTTP protokolem.

Technologie jazyka Dart umožňují spustit kód dvěma způsoby. První z nich je určen pro mobilní a desktopové aplikace. Dart obsahuje virtuální stroj s JIT[3] kompilací, který se používá především ve fázi vývoje. Tento způsob umožňuje tzv. inkrementální rekompilaci jejíž výhodou je funkcionalita *hot reload* či nástroje *DevTools* pro aktuální metriky ladění. Dále AOT[4] kompilátor pro tvorbu strojového kódu, který se využívá při sestavování aplikace pro použití v produkci. Druhý způsob se týká webové platformy, kde Dart umožňuje kód přeložit do jazyků JavaScript nebo WebAssembly. I zde Dart využívá rozdílné techniky pro ladění kódu a následnou produkci. Oba tyto způsoby jsou stejné v tom že potřebují *běhové prostředí Dart*. Toto prostředí se stará o správu paměti pomocí garbage collector, vynucení kontroly typů a správu vláken.

3.2 Flutter

Open source framework Flutter [7] slouží pro tvorbu uživatelských rozhraní. Využívá jazyk Dart a je také vyvíjený společností Google. Poprvé zveřejněn byl

[3] Just in time
[4] Ahead of time

```
class Point {
  final double x;
  final double y;

  const Point(this.x, this.y);

  bool get isInsideUnitCircle => x * x + y * y <= 1;
}
\label{code:dart}
```

Zdrojový kód 1: Ukázka kódu v jazyce Dart

v květnu 2017. Google sám ho využívá v aplikacích jako je Google Play nebo Google Earth [8]. Mezi známé aplikace třetích stran používajících Flutter patří Bakaláři Software nebo hra PUBG Mobile [8].

Používá vlastní vykreslovací jádro, které pixely přímo vykresluje na obrazovku. Toto je podstatný rozdíl oproti řadě jiných frameworků, které se spoléhají na vykreslovací jádro dané platformy. Tento přístup umožňuje mít identicky vypadající uživatelské rozhraní napříč všemi podporovanými platformami.

Základní komponentou je *widget*. Ten se může skládat z dalších widgetů a vytvářet tak stromovou strukturu. Celé uživatelské rozhraní je poskládáno z widgetů. Flutter poskytuje dva typy předdefinovaných widgetů podle dvou designových systémů – Material Design (viz kapitola 3.3 Material Design) widgety a Cupertino widgety (respektující Human Interface pravidla [9] od společnosti Apple). Přestože oba mají svoji primární platformu, Flutter je umožňuje používat libovolně. Programátor si samozřejmě může definovat widgety vlastní.

Pro rozložení prvků na stránce (vytvoření *layout*) se taktéž používají widgety, přestože nejsou při zobrazení viditelné. Základními widgety pro tvorbu layoutu jsou `Row`, `Column` a `Container`. Pomocí `Container` můžeme ostatním widgetům přidávat odsazení, ohraničení, transformaci atd. Tyto widgety slouží pro specifickou či nízkoúrovňovou tvorbu layoutu. Můžeme využít i specializované widgety jako `GridView` pro tvorbu mřížky nebo `ListView` pro rolovatelný seznam. Nejvíce specifické widgety jako `BottomNavigationBar` či `AppBar` zajistí dodržení pravidel stanovených systémem Material Design a umožní programátorovi velmi jednoduchou implementaci těchto komponent.

3.3 Material Design

V březnu 2025 je Material Design verze 3 [10] nejnovějším open source designový systém od společnosti Google. Systémem v tomto případě mám na mysli soubor pravidel pro uživatelské rozhraní, konkrétní komponenty i barevné provedení. Všechny tyto prvky jsou vytvořeny zkušenými designéry s respektem pro psychologický vliv uživatelského rozhraní na člověka.

Označení verze 3 napovídá, že v minulosti již proběhly aktualizace tohoto

systému. Tento krok dává smysl vzhledem k vyvíjejícím se trendům v oblasti designu. Aktualizace mohl uživatel pozorovat v operačním systému Android nebo v aplikacích od Google, jelikož Google tento systém používá téměř všude.

Použití tohoto systému má ve Flutteru řadu výhod. Především většina widgetů ze systému Material Design je již implementována a použití programátorem je velmi jednoduché. Programátor nemusí často řešit rozmístění jednotlivých prvků v konkrétní komponentě ani správnou adaptaci na velikost zařízení. Běžnou součástí aplikací jsou ikony. Ikony z Material Designu jsou ve Flutteru k přímému použití bez specifického nastavení. Programátor nemusí řešit žádné externí SVG soubory. Stejně to platí i pro použití fontů písem.

3.4 SQLite

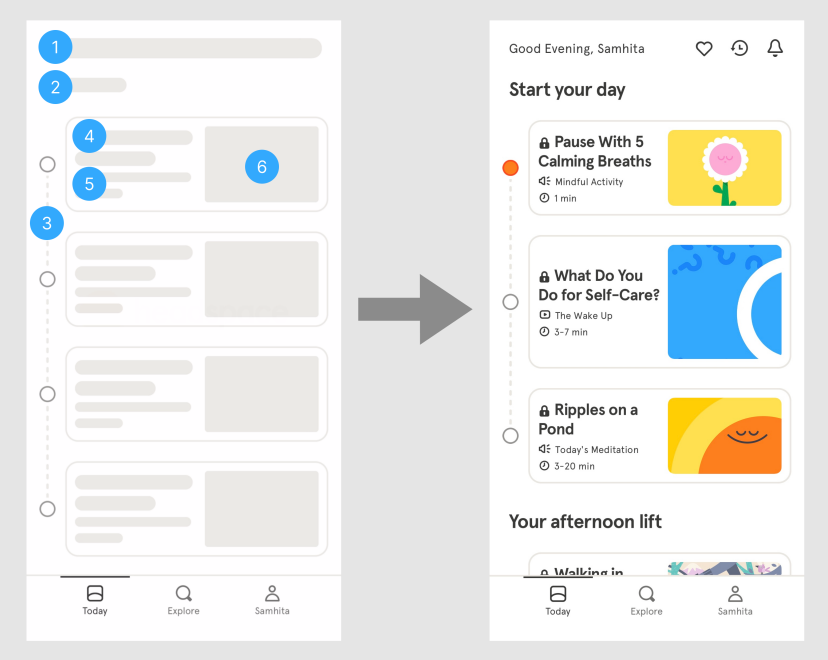
SQLite [11] je velmi jednoduchá a rychlá relační databáze, která je uložena v jediném souboru. Vývoj SQLite byl zahájen v roce 2000. Obvykle jsou všechny potřebné závislosti již zabudovány v zařízeních, ať už jde o mobilní telefony nebo desktopové operační systémy. Je zdarma k užití pro libovolné účely. Přesto poskytuje plnohodnotnou SQL implementaci. Zajímavostí je, že onen soubor je multiplatformní. Nevadí mu přenos mezi 32bitovými a 64bitovými systémy nebo architekturami big-endian a little-endian.

3.5 Balíčky

Přestože balíčky tvoří závislosti projektu na ostatních okolnostech a programátor se musí spoléhat na jiné programátory, že je budou udržovat aktuální, je nemožné se jim v dnešním době vyhnout. Dart a Flutter má na výběr z široké veřejné knihovny balíčků. Balíčky je nutné do projektu importovat. Celý jejich seznam je dostupný v souboru `pubspec.yaml`. Přidání balíčku do projektu se provádí zapsáním do souboru `pubspec.yaml` a spuštěním příkazu `flutter pub get` nebo rovnou příkazem `flutter pub add <název_balíčku>`.

- **Drift** [12] je balíček poskytující rozhraní pro práci s SQLite databází. Jeho velkou výhodou oproti ostatním implementacím SQLite pro Dart je možnost multiplatformnosti. Funguje na všech platformách od Androidu přes Windows až po webové rozhraní (na rozdíl od knihovny sqflite, která je zmíněna v oficiálním návodu). Dále poskytuje velmi jednoduché API pro vytváření dotazů nad databází bez použití jazyka SQL. Programátor vytváří strukturu databáze v jedníném souboru pomocí jazyka Dart. Na základě tohoto souboru si Drift vygeneruje interní soubory.

- **GoRouter** [13] je deklarativní balíček pro organizaci navigace v rámci aplikace (tzv. *routování*). Funguje na principu URL adres. Umožňuje v adrese předávat parametry, zajistit přesměrování na základě práv uživatele či použití *vnitřních navigátorů* nejčastěji pro navigaci, která zůstává neustále viditelná skrz více obrazovek. Pro přechody umožňuje nastavit vlastní libovolné animace. Autory jsou přímo oficiální vývojáři frameworku Flutter.

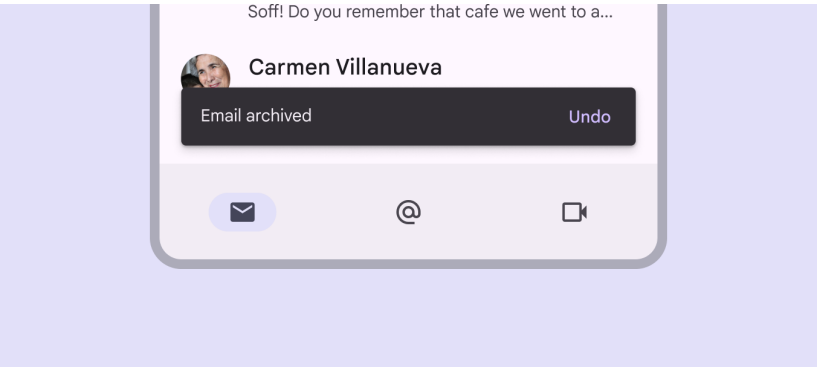


Obrázek 6: Skeleton loading a načtená obrazovka – převzato z [15]

- **Skeletonizer** [14] zjednodušeným způsobem poskytuje funkcionalitu označovanou jako *skeleton loading*. Používá se při načítání obsahu na stránce a zobrazuje hrubý náhled rozložení prvků na stránce (znázorněno na Obrázku 6). Vše je doplněno vhodnou animací, aby uživatel ihned poznal, že se obrazovka načítá. Zjednodušení použitím tohoto balíčku spočívá v tom, že programátorovi stačí již definovaný widget „obalit“ widgetem `Skeletonizer` z tohoto balíčku a o zbytek práce se postará balíček.

- **Another Flushbar** [16] poskytuje vylepšenou komponentu `Snackbar` ze systému Material Design. Účelem komponenty je krátce informovat uživatele o akci, která byla právě provedena (například akce úspěšné archivace e-mailu je znázorněna na Obrázku 7). Obvykle se objevuje ve spodní části obrazovky. Vylepšení spočívá v možnosti většího přizpůsobení – přidání ikony, změny barvy či libovolného umístění. Já tento balíček preferoval i z důvodu správného zobrazování při otevřeném dialogovém okně.

- **Syncfusion Flutter Charts** [18] je rozsáhlý balíček pro tvorbu grafů. Umožňuje vytvářet všechny druhy grafů – sloupcové, spojnicové, prstencové a další (autoři uvádí více než 30 druhů). Grafy lze dále přizpůsobovat – výběr animace, úprav jednotek na osách, výběr barev, možnost zobrazit legendu či podrobné informace daného bodu v grafu po přejetí myší. Přes širokou škálu úprav je základní použití velmi jednoduché. Bonusem je podrobná dokumentace s ukázkami všech typů grafů a jejich zdrojových kódů.



Obrázek 7: Komponenta Snackbar – převzato z [17]

- **File Picker** [19] dává programátorovi možnost využít nativní průzkumník souborů k výběru složky nebo konkrétních souborů pro jejich další zpracování v aplikaci. Je možnost soubory filtrovat či umožnit výběr více z nich najednou. Základní funkcionalitu poskytuje na libovolné platformně s velmi jednoduchou implementací. V mém případě jsem ho použil pro výběr složky, kam si uživatel chce uložit exportované soubory.

4 Programátorská příručka

Multiplatformní vývoj přináší jistá specifika (nutnost zvolit vhodné technologie, případnou optimalizaci zobrazení či zúžený výběr knihoven). Z hlediska architektury či samotného procesu programování je postup z většiny totožný jako u programování pro jednu platformu. Zvolená technologie často sama navádí k použití vhodných prvků.

Mnou navržené a vytvořené aplikaci jsem dal název Budget Buddy. Je určena pro mobilní telefony, tablety i desktopová zařízení. Testování probíhalo na operačních systémech Android a Windows. Pro ostatní operační systémy není zaručena plná funkcionalita, i když z podstaty frameworku by neměly nastat problémy.

Pro vývoj aplikace jsem použil textový editor Visual Studio Code s řadou rozšíření. Část práce specifické pro platformu Android jsem realizoval v prostředí Android Studio. Především se jednalo o úlohy typu aktualizace Software Developement Kit či migrace na novější verzi Kotlin gradle, kde Android Studio poskytuje jednodušší rozhraní a programátora provede celým procesem.

4.1 Architektura aplikace

Pro multiplatformní vývoj neexistuje doporučená architektura už z podstaty věci širokého záběru různých zařízení. Je možné se držet dlouho známých architektur, které se používají v různých systémech (*Model-View-Controller* či *Mode-View-ViewModel*) nebo se řídit podle zvyklostí daného frameworku (ve Flutteru se

jedná například o *Bloc* [20] architekturu). Já se ve své aplikaci držel architektury Model-View-ViewModel, jelikož s ní mám z minulosti zkušenosti a pro moji malou aplikaci je plně dostačující.

Model-View-ViewModel (zkráceně MVVM) je architektura, která odděluje v aplikaci uživatelské rozhraní, logiku a datovou část [21]. Ve frameworku Flutter je velmi dobře podporována, jednak z hlediska komponent frameworku, tak i oficiální dokumentace na architekturu odkazuje jako vhodný příklad návrhu aplikace. Rozdělení aplikace je do následujících tří detailněji popsaných celků.

- **Model** je částí zapouzdřující data, se kterými aplikace pracuje. Udává strukturu poskytovaných dat i metody pro jejich získání objekty ve ViewModel.
- **View** je zodpovědné za strukturu a zobrazení uživatelského rozhraní. Neměl by obsahovat žádnou logiku aplikace. Vstupy které přijme od uživatele předává ke zpracování na ViewModel.
- **ViewModel** je prostředníkem mezi View a Model. Do View poskytuje data a případně je upraví do vhodného formátu pro zobrazení. Reaguje na zprávy od View, které představují vstupy uživatele. Vlákno pro uživatelské rozhraní by neměl blokovat, zlepší se tím uživatelský zážitek.

V mé konkrétní implementaci je kód strukturován do složek podle MVVM architektury – `model`, `repository`, `view` a `view_model`. Ve `view` je jemnější rozdělení na obrazovky `screens` a jednotlivé widgety `widgets`. Složka `repository` slouží pro databázi a operace nad ní. Složka `utils` má v sobě definované jednoduché funkce používané napříč projektem (převod kódu ikony na Ikonu, zkrášlení číselného formátu a další) a složka `theme` spravuje vizuální téma aplikace.

4.2 Uchovávání stavu

Uchování stavu jednotlivých obrazovek a jiných prvků je velmi důležité pro uživatelský komfort. Běžně se uživatel překlikne nebo přeskakuje mezi obrazovkami a není žádoucí, aby po každé byla obrazovka ve výchozím stavu (např. aby bylo nutné opět scrollovat v seznamu). Použití GoRouter za programátora řeší i tento problém. Není nutné nic nastavovat a stačí řídit se dokumentací GoRouter. Po zavedení do aplikace již uchování stavů funguje přesně tak, jak je pro uživatelský komfort vhodné. V první řade bylo nutné vytvořit aplikaci konstruktorem `MaterialApp.router`. Do parametru `_router` jsem vložil konfiguraci GoRouter (viz Zdrojový kód 2). V konfiguraci je uvedena cesta a obrazovka, která se při jejím zadání zobrazí.

4.3 Implementace responzivního designu

Responzivní design má zajistit, že aplikace se bude vhodně zobrazovat na libovolně velkém zařízení. Vyskytuje se především u webových technologií z důvodu

```
// GoRouter konfigurace
final _router = GoRouter(
  routes: [
    GoRoute(
      path: '/',
      builder: (context, state) => HomeScreen(),
    ),
  ],
);

// konstruktor aplikace
class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp.router(
      routerConfig: _router,
    );
  }
}
```

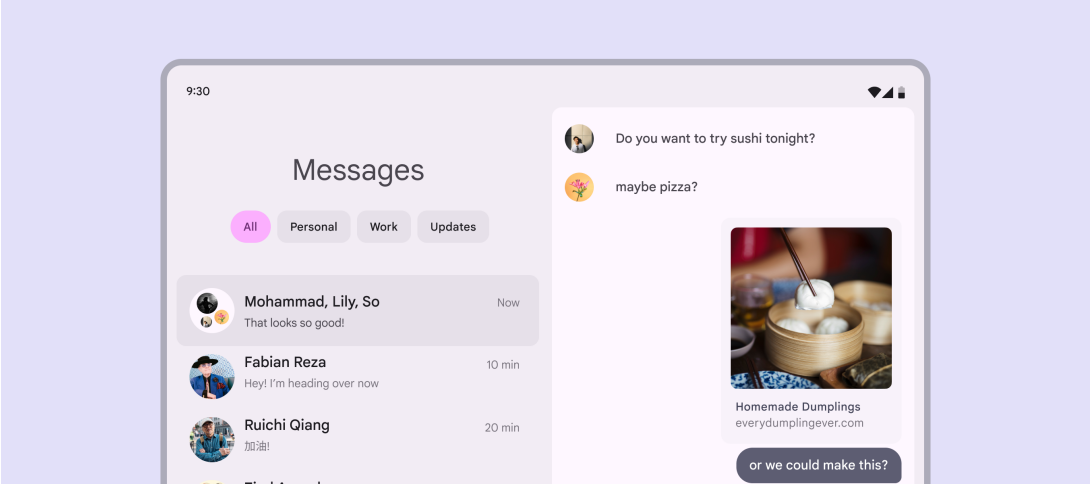
Zdrojový kód 2: Ukázka použití GoRouter

běžného používání na různých typech zařízení. Stejná problematika je u multiplatformních aplikací.

Podle velikosti obrazovky je nutné vhodně zvolit z určitých komponent, které slouží pro stejný účel, ale jsou jinak zobrazené. V této aplikaci se jedná například o navigaci. Pro různou velikost obrazovky jsem použil různé komponenty – navigation bar, navigation rail a navigation drawer (viz Obrázek 8). Dále je nutné vhodně zobrazit komponenty v hlavním těle obrazovky. Řazení komponent na menším zařízení jsem zvolil pod sebe do sloupce, na větším naopak vedle sebe v řádku. Je dostupná i komponenta `GridView` vytvářející tabulku. Avšak není vhodná pro tvorbu layoutu. Material Design poskytuje i speciální kanonické rozložení (viz Obrázek 9), které je vhodné ve spojení se zobrazením seznamu. V této aplikaci by bylo vhodné pro obrazovku transakcí. Implementace pro framework Flutter zatím není dostupná, tudíž není možné ho použít. Dialogová okna jsem podle doporučení pro menší zařízení zvolil celoobrazovková, zatímco na větších obrazovkách jsou plovoucí (viz Obrázek 14).

Framework Flutter poskytuje komponenty vhodné pro responzivní design, ale na programátorovi je ponecháno kdy jaké zvolí, včetně celé implementace. Zatím není dostupná žádná komponenta pro tvorbu responzivního rozložení, které by umožnila v jediném kódu pro uživatelské rozhraní rozhodnout do jakého widgetu kód „obalit“.

Obrázek 8: Navigační prvky – převzato z [10]



Obrázek 9: Kanonické rozložení – převzato z [10]

Obrázek 10: Ikona aplikace

Nikdo nesmí být svévolně zatčen, držen ve vazbě nebo vyhoštěn do vyhnanství.

Nikdo nesmí být svévolně zatčen, držen ve vazbě nebo vyhoštěn do vyhnanství.

Obrázek 11: Font Roboto (nahoře) a Poppins (dole)

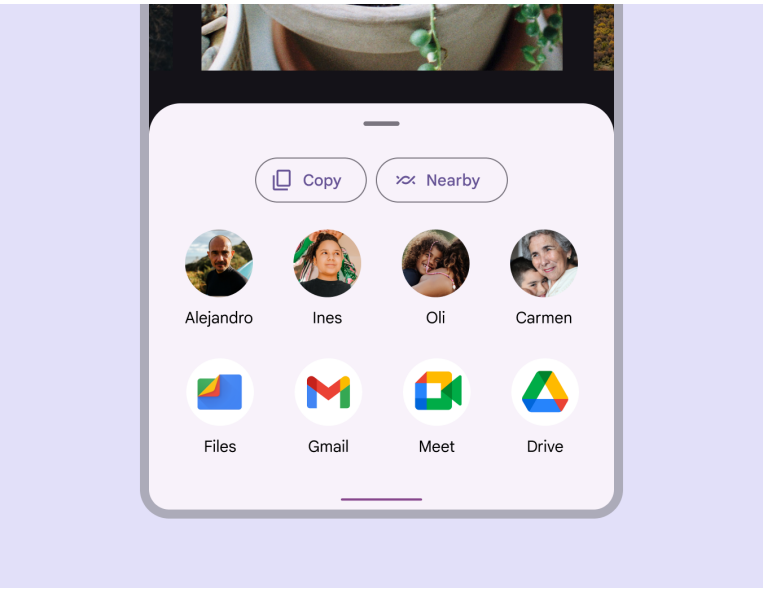
4.4 Vizuální část aplikace

Vizuální část aplikace je z pohledu uživatele velmi důležitá. Řadí se do ní barvy v aplikaci, ikona aplikace, použité fonty atd. Primární barvou aplikace jsem zvolil tmavě tyrkysovou. Je nastavená jako výchozí barva pro barevné téma a použil jsem ji i pro ikonu aplikace. Ikona aplikace (viz Obrázek 10) byla vytvořena v nástroji Figma. Byly v ní použity ikony z Google Fonts. Jako font aplikace jsem zvolil font *Poppins*, který je podobný doporučenému fontu *Roboto* ze systému Material Design, ale je více zaoblený. Fonty jsou vidět na Obrázku 11.

4.5 Zajímavosti při tvorbě práce

Tvorba aplikace oproti návrhu místy přinese neplánované komplikace, které je nutné řešit, případně redefinovat požadavek, který je způsobil. I v tomto projektu jsem se s touto výzvou setkal.

- **Problém s výběrem SQLite balíčku** nastal hned ze začátku programování aplikace. Při postupování podle dokumentace [22], která ukazuje práci s balíčkem sqflite, jsem při čtení dokumentace balíčku zjistil, že balíček podporuje pouze Android, iOS a macOS. Vzhledem k zamýšlenému provozu i na jiných platformách bylo nutné najít alternativu. Objevil jsem balíček Drift, který podporuje všechny platformy. Navíc přinesl i bonus v podobě kvalitní API pro dotazování nad databází bez použití jazyka SQL.
- **Nekompletní podpora všech komponent Material Design**. Přestože framework Flutter i systém Material Design mají stejného vývojáře



Obrázek 12: Komponenta BottomSheet – převzato z [10]

firmu Google, není jejich propojení 100%. Flutter většinu komponent implementuje bez problémů, ale některé nejsou vůbec podporovány, případně je jejich použití omezené. Jako příklad uvedu komponentu `BottomSheet`, která je vidět na Obrázku 12 a slouží ke zobrazení doplňkového obsahu a akcí. Případně `LayoutBuilder` sloužící k tvorbě responzivního designu. Programátor musí problémy, které běžně řeší tyto komponenty, nahradit nebo jejich řešení implementovat sám.

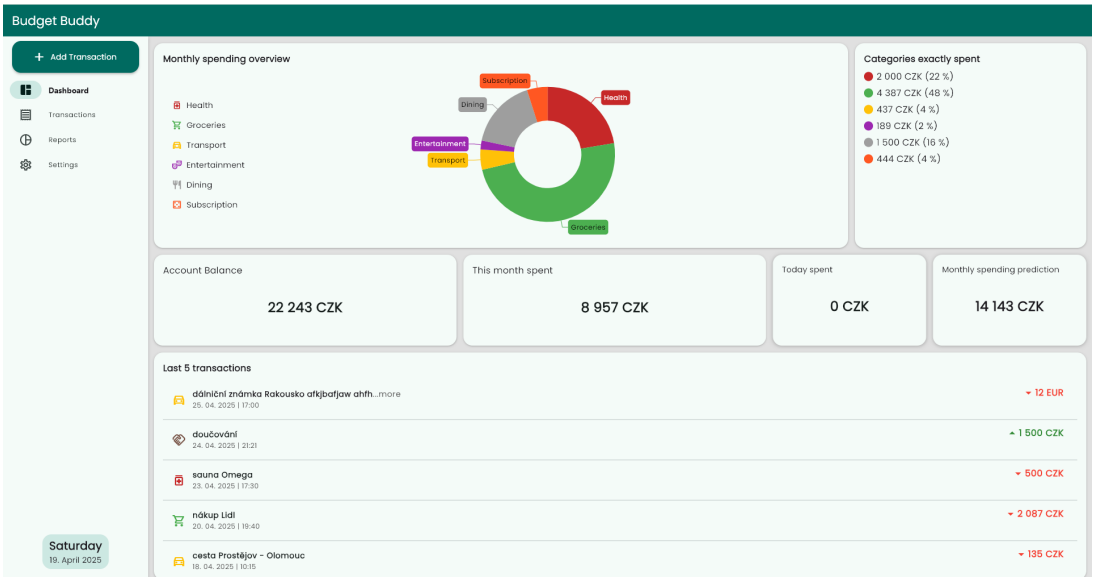
- **Odstranění kategorií či měn** přináší navazující problém, jak naložit s transakcemi, které mají danou měnu nebo kategorii přiřazenou. Existují dvě možná řešení – navázané transakce přenést pod jinou kategorii/měnu podle volby uživatele nebo navázané transakce odstranit. Kvůli jednodušší implementaci jsem zvolil druhou možnost.

5 Uživatelská příručka

Aplikaci Budget Buddy pro správu financí lze nainstalovat na operační systém Android verze 5.0 a vyšší a Windows 10 a vyšší se zárukou bezproblémové funkčnosti podle návodu. Pro operační systémy iOS, macOS a Linux je nutné aplikaci nejdříve sestavit a následně nainstalovat (tento postup je ponechán na zdatném uživateli). V návodu jsou uvedeny jen ony dvě platformy z důvodu testování.

5.1 První spuštění

Při prvním spuštění se zobrazí rovnou úvodní obrazovka *Dashboard*. Není nutné se přihlašovat, aplikace je nyní určena pouze pro jednoho uživatele a běží lokálně.



Obrázek 13: Obrazovka *Dashboard* na desktopovém zařízení

Pro první spuštění jsou předpřipravené základní kategorie transakcí a nejběžnější měny (obojí je možno upravit).

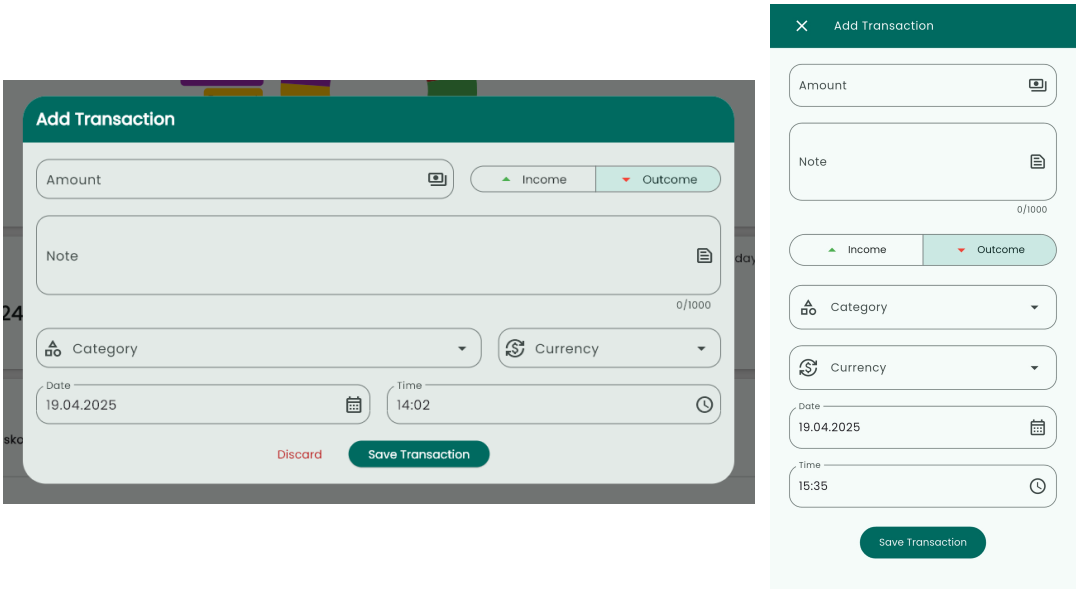
5.2 Úvodní obrazovka

Úvodní obrazovka (viz Obrázek 13) je centrem dění pro uživatele. Poskytuje základní přehled o finanční situaci. Konkrétně zobrazuje graf útraty podle kategorií (v aktuálním měsíci), stav účtu, dnešní útratu a posledních pět transakcí. Pro zařízení s větší obrazovkou je rozšířena o legendu grafu a další zajímavé číselné údaje o stavu účtu. Úvodní obrazovka není uživatelsky přizpůsobitelná.

5.3 Přidání transakce

Transakce se přidává tlačítkem označeným ikonou plus, na větším zařízení i doplňujícím popiskem. Na menším zařízení se tlačítko nachází v pravé dolní části obrazovky nad navigační lištou jako tzv. *floating action button*. Na větším zařízení je umístěno před prvním prvkem v navigační liště na pravé straně obrazovky. Po stisknutí se objeví dialogové okno, které je vidět na Obrázku 14, kde uživatel může vybrat z následujících možností:

- **Částka** – zadává se hodnota transakce jako číslo. Desetinná čísla jsou povolená jak s čárkou i s tečkou (čárku si aplikace sama převede na tečku).
- **Poznámka** – textový doplněk transakce o délce maximálně 1000 znaků. Poznámka není povinná.
- **Typ** – výběr ze dvou možností zda-li jde o útratu nebo příjem.



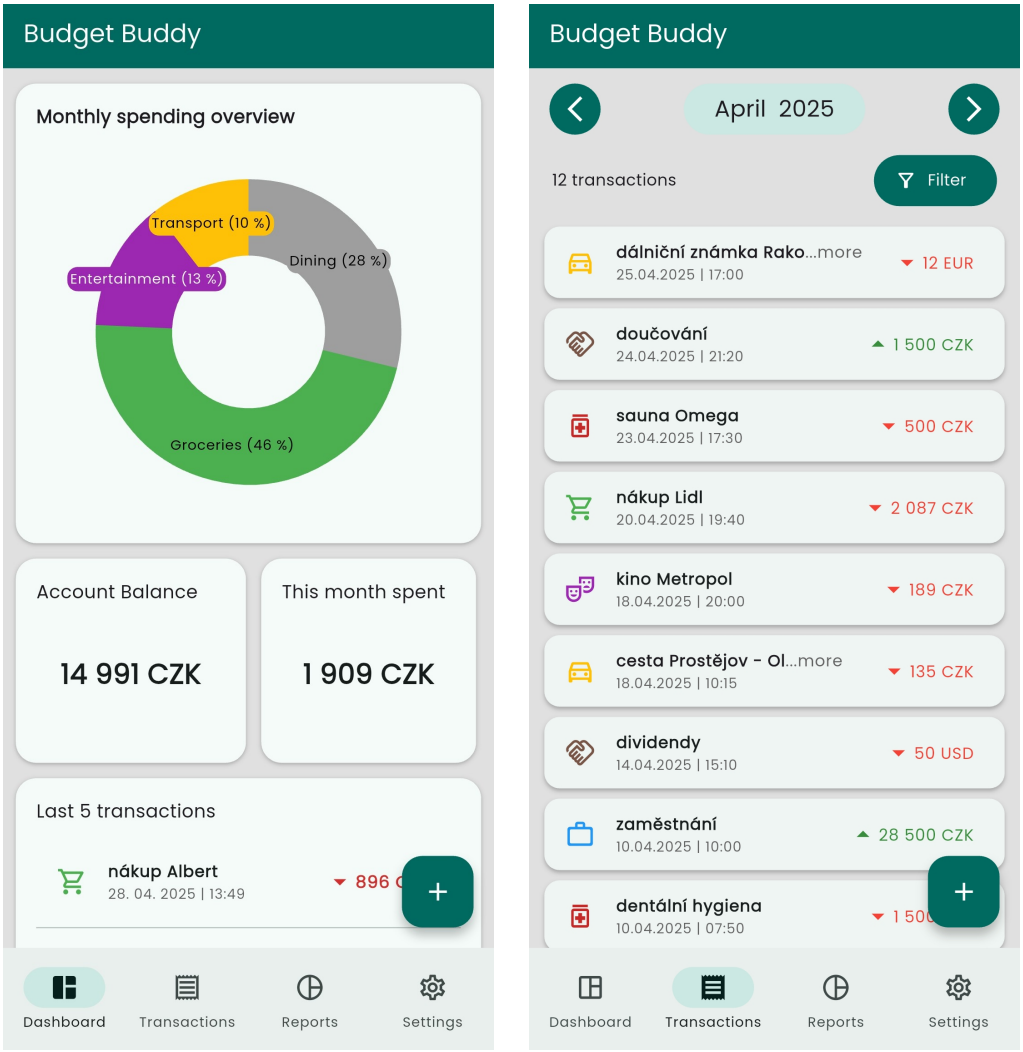
Obrázek 14: Dialogové okno pro přidání transakce na desktopovém zařízení (vlevo) a mobilním telefonu (vpravo)

- **Kategorie** – kategorie se kterou bude transakce spojena. Výběr se zobrazí ve vyjíždějícím seznamu po kliknutí.
- **Měna** – měna dané transakce. Pokud se jedná o měnu jinou než Česká koruna, bude provedena konverze podle aktuálně nastaveného kurzu.
- **Datum** – jaký den byla transakce provedena. Předvyplněno na aktuální datum.
- **Čas** – v jakou denní dobu byla transakce provedena. Předvyplněno na aktuální čas.

Vyplněnou transakci je nutné tlačítkem *Save Transaction* uložit jinak nebude zápis proveden.

5.4 Zobrazení a úprava transakcí

Výpis transakcí a jejich úprava se provádí na druhé obrazovce s názvem *Transactions*. Výpis je proveden v seznamu, kde je přímo vidět kategorie, měna, částka, typ, datum a případná poznámka (viz Obrázek 15). Po kliknutí na transakci se objeví dialogové okno a je možné všechny parametry transakce upravit (viz Obrázek 16 vlevo), případně transakci odstranit. Transakce je možné filtrovat dvěma způsoby (viz Obrázek 16 vpravo). Můžeme nechat zobrazit jen transakce ve specifickém období (den, týden, měsíc, rok či libovolný rozsah), k čemuž slouží prostřední tlačítko v horní části zobrazující aktuálně zvolené období. Šipky na stranách umožňují přesouvat se o zvolené období dozadu či dopředu. Dále můžeme vybírat ze všech parametrů, které jsou transakci přiřazené (kategorie, měna,



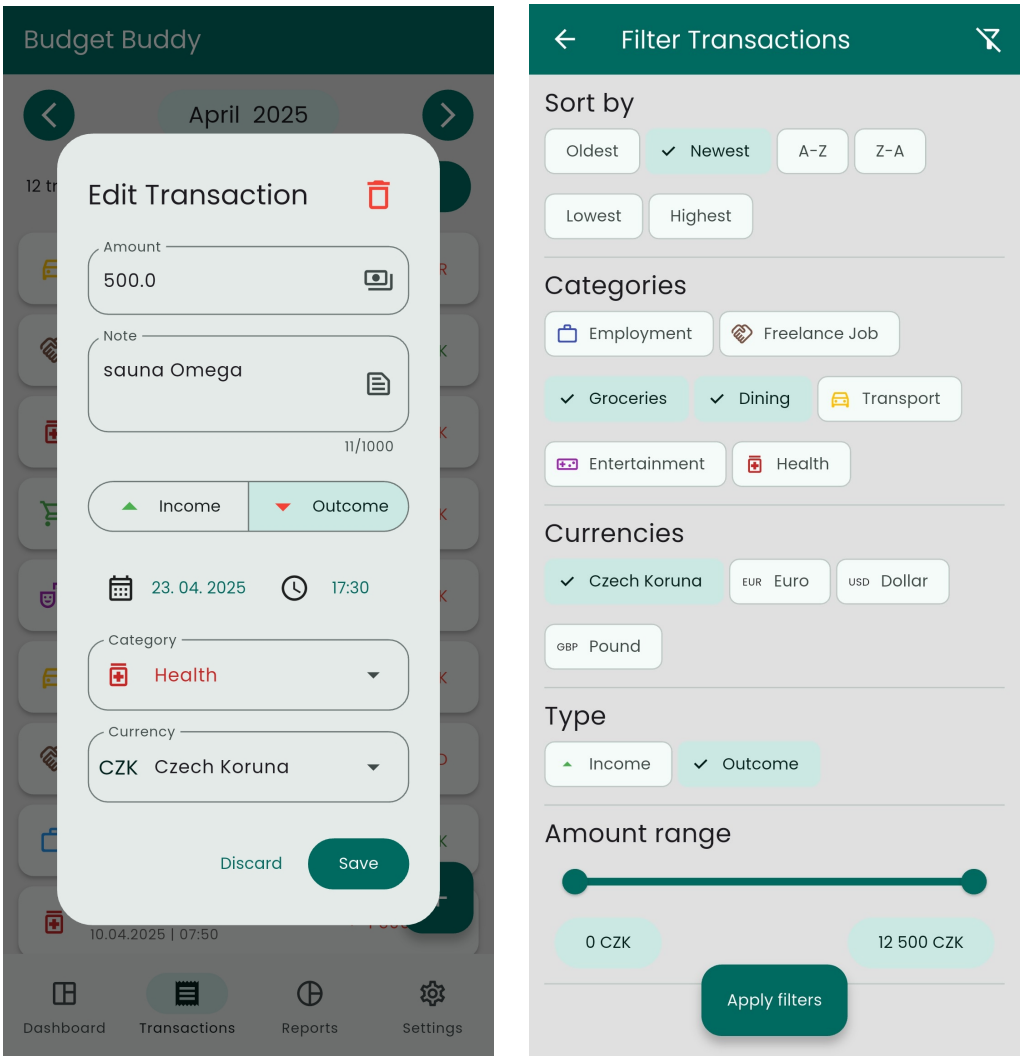
Obrázek 15: Obrazovky *Dashboard* a *Transactions* na mobilním telefonu

typ atd.). Obrazovka s výběrem filtrů se otevře při stisknutí tlačítka *Filter* s ikonou trychtýře. Při použití více filtrů se transakce zobrazí ve výpisu pokud splňuje alespoň jednu podmínku. Zvolené filtry je nutné potvrdit tlačítkem *Apply filters*. Aktivní filtry znázorňuje červený odznak u tlačítka filtrace.

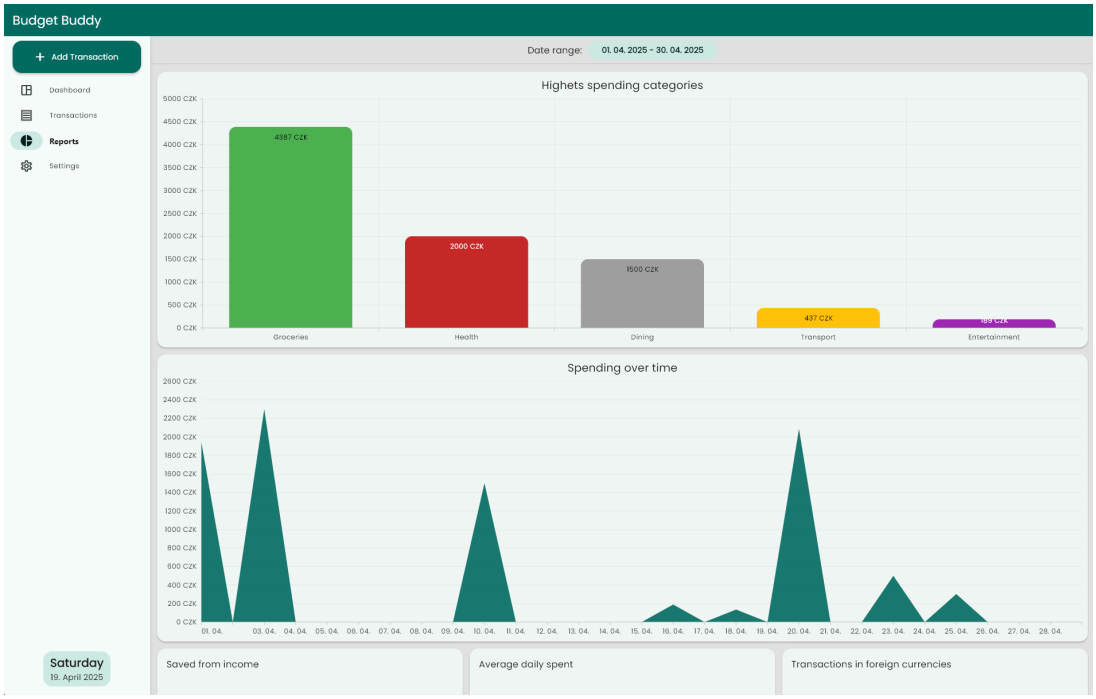
5.5 Využití reportů

Grafy a důležité číselné údaje se nachází na obrazovce *Reports* (viz Obrázek 17). Je zde na výběr z pěti přednastavených možností. Které údaje chce uživatel vidět se dá nastavit tlačítkem *Manage reports* pod posledním reportem. Na výběr je z pěti možností:

- **Top categories** – zobrazí sloupcový graf s pěti kategoriemi, ve kterých je největší útrata ve zvoleném období.



Obrázek 16: Úprava a filtrace transakcí na mobilním telefonu

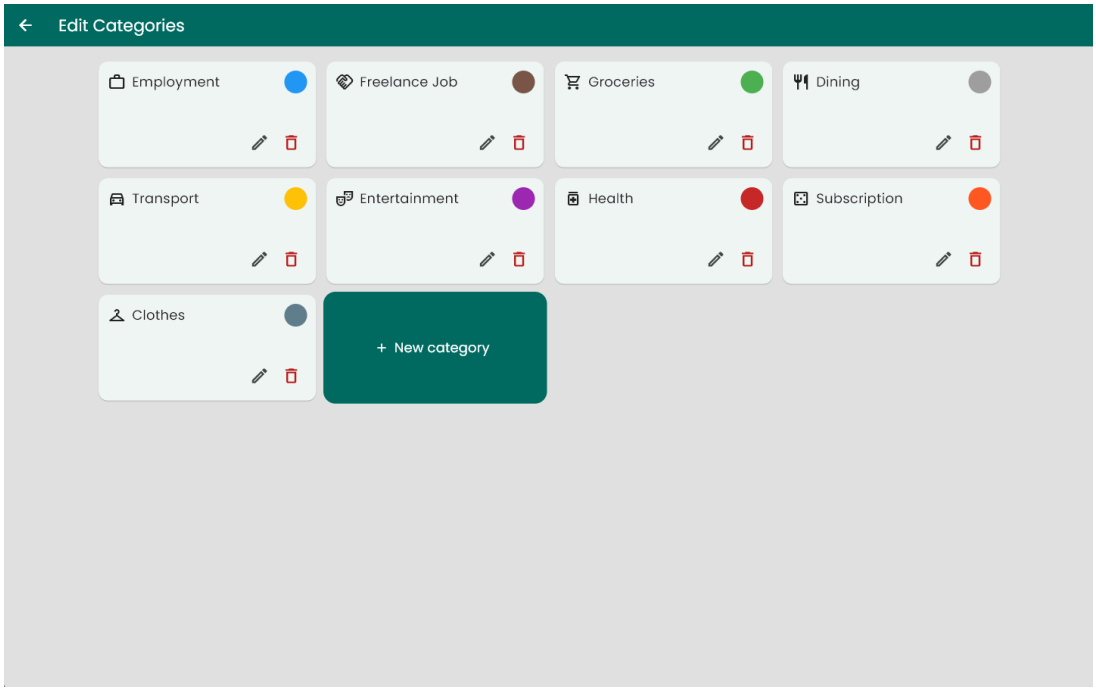


Obrázek 17: Obrazovka *Reports* na desktopovém zařízení

- **Spent during time** – zobrazí plošný graf s vývojem útraty v průběhu daného období. Graf je interaktivní a při najetí pohybu kurzorem v grafu se zobrazí hodnota pro konkrétní datum.
- **Interesting numbers** – zobrazí tři buňky se zajímavými hodnotami ze zvoleného období. Konkrétně se jedná o procento ušetřených peněz z příjmu, o průměrnou denní útratu a o procento transakcí v jiné měně, než českých korunách. Podle velikosti zařízení jsou buňky řazeny ve sloupci nebo řádku.
- **Category ratio** – zobrazí dvě buňky s kruhovými grafy. První zobrazuje poměr kategorií v příjmech a druhý poměr kategorií ve výdajích. Při použití na největším zařízení se zobrazí interaktivní legenda.
- **Income / Outcome numbers** – zobrazí tři buňky s číselnými údaji. Konkrétně se jedná o součet všech příjmů, o součet všech výdajů a celkový stav účtu. Podle velikosti zařízení jsou buňky řazeny ve sloupci nebo řádku.

5.6 Nastavení aplikace

Poslední obrazovkou dostupnou z navigační lišty je nastavení (viz Obrázek 19). Umožňuje nastavit jak prostředí po stránce funkcí, tak i po stránce vzhledu. Taktéž zde uživatel nalezne informace o aplikaci a možnost smazání všech dat.



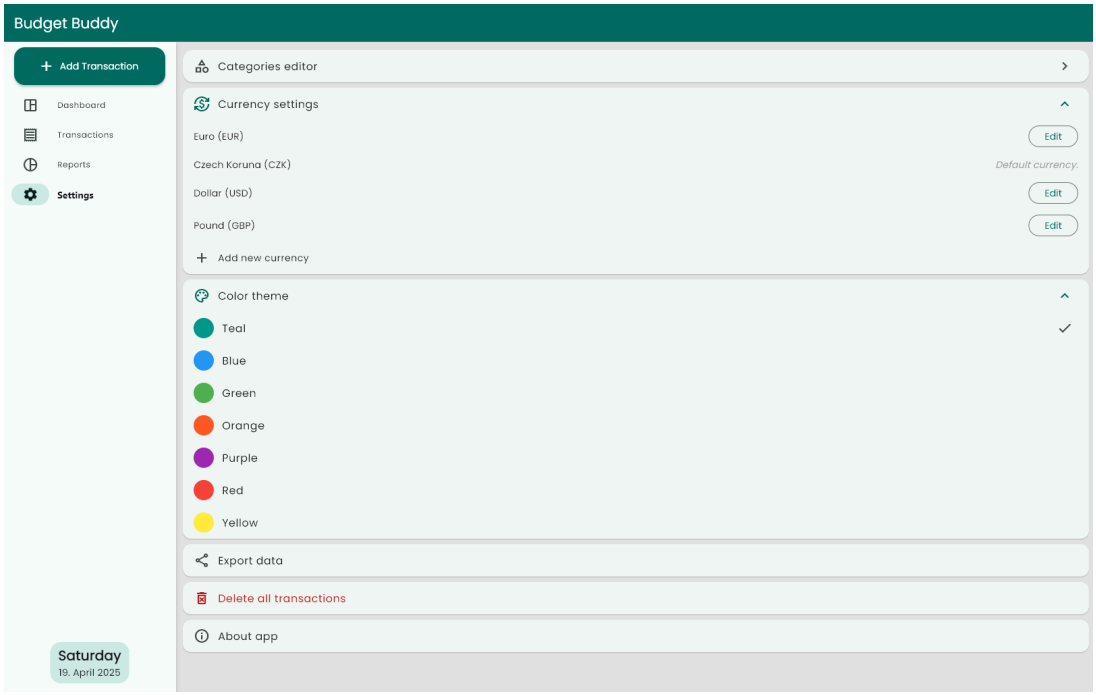
Obrázek 18: Editor kategorií na desktopovém zařízení

5.6.1 Úprava kategorií

Přidávat, mazat i upravovat kategorie je možné v záložce *Categories editor* (viz Obrázek 18). Otevře se nová obrazovka s výpisem všech kategorií formou karet. U každé karty je znázorněna ikona, název a barva. Ikona tužky otevře dialogové okno s možností upravit nastavení kategorie (změny je nutné uložit tlačítkem *Save*). Ikona popelnice po potvrzení v dialogovém okně nevratně smaže danou kategorii i jí náležící transakce. Plovoucím tlačítkem s ikonou plus se vytváří nová kategorie s uživatelem definovanými parametry.

5.6.2 Přizpůsobení měn

Po kliknutí na kartu *Currency settings* se rozbalí seznam všech měn. V seznamu je vidět název a zkratka měny. Na pravé straně každé měny je tlačítko *Edit*, které vyvolá dialogové okno pro editaci měny. V něm je možné zvolit název, směnný kurz a zkratku. Doporučený formát pro zkratku je ISO 4217 [23], jelikož zaručuje kompatibilitu s aktualizací kurzu z České národní banky. Příklady formátu jsou USD pro americký dolar, CZK pro českou korunu nebo EUR pro euro. Aktualizace směnného kurzu se provádí tlačítkem i ikonou šipek v kruhu. Směnné kurzy jsou čerpány ze stránek České národní banky [24]. Změna kurzu nemá vliv na v minulosti uložené transakce.



Obrázek 19: Obrazovka *Settings* na desktopovém zařízení

5.6.3 Přizpůsobení vzhledu

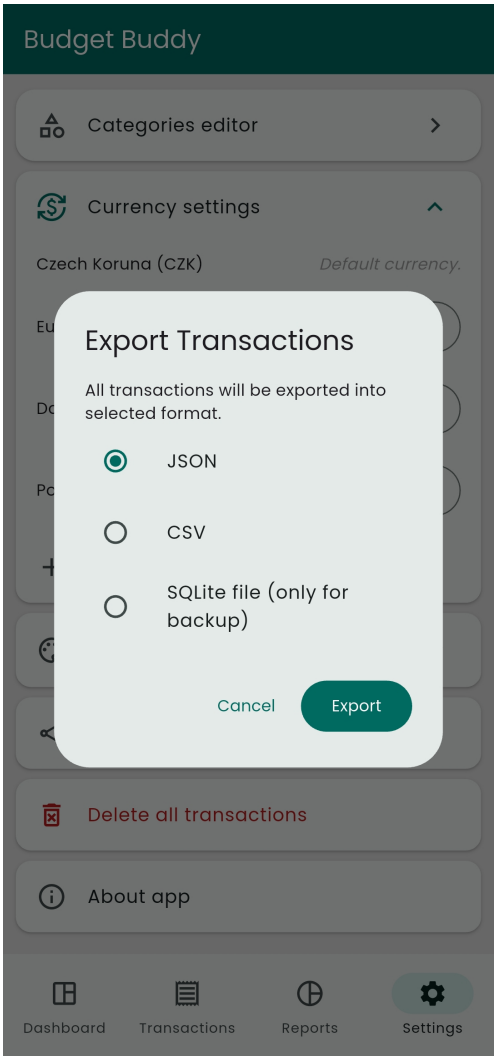
Vzhled je možno přizpůsobit rozbalením karty *Color theme*. Výběr je ze sedmi barev, které jsou pojmenované i s náhledem konkrétní barvy. Konkrétní barvy reprezentují jednotlivá barevná témata (jedna barva je základem tématu a z ní se odvíjí barvy ostatní). Výběr tohoto tématu změní celý nádech aplikace od výrazných barev až po jemné barvy v pozadí. Změna proběhne ihned a není nutné ji potvrzovat.

5.7 Export dat

Export dat se nachází pod položkou *Export data* (viz Obrázek 20). Je možný ve třech formátech – CSV, JSON a SQLite. První dva formáty slouží především pro import dat do jiných aplikací nebo pro libovolné zpracování uživatelem. Jsou široce podporované a data se dají interpretovat i bez dalšího zpracování. Poslední formát slouží pro zálohu dat. Jde o soubor s kompletní SQLite databází. Pro export dat je možné si vybrat libovolné umístění na právě používaném zařízení.

6 Možná rozšíření aplikace

Osobně si myslím, že se nikdy nedá o softwaru prohlásit, že by nepotřeboval vývoj a není možné ho vylepšit. Už vzhledem k vývoji technologií a proměně uživatelských požadavků je nutné aplikaci udržovat aktuální. Se změnou požadavků



Obrázek 20: Dialogové okno pro export dat na mobilním telefonu

se pojí rozšíření o nové funkce a nebo naopak odstranění funkcí nepoužívaných.

1. **Možnost založit více finančních účtů** nabízí většina existujících řešení, proto se z mého pohledu jedná o logický krok. Tento krok by umožnil uživateli si oddělit různé účty, případně si je libovolně organizovat. Na druhou stranu se jedná o další potenciální komplikaci pro uživatele a bylo by nutné zajistit vhodné přepínání účtů i správné zobrazení momentálně zvoleného účtu.

2. **Překlad aplikace do jiných jazyků** umožní zpřístupnění pro uživatele ze zahraničí. Překlad aplikace není z technického hlediska náročný a je možné ho provádět postupně. Samotné přeložení výrazů je za pomocí překladačů postavených na velkých jazykových modelech snadnou a levnou záležitostí.

3. **Import dat z jiných aplikací** je funkcionalitou, která může přimět uživatele k přechodu na tuto aplikaci. Většina aplikací pro správu financí nabízí export dat do CSV nebo JSON formátu. I když různé aplikace mají odlišné struktury dat, pro nejběžnější aplikace by bylo možné vytvořit konkrétní proces importu.

4. **Umožnit výběr hlavní měny** namísto momentální pevně zvolené České koruny. Aplikace nyní kvůli sjednocení transakcí v zahraničních měnách používá Českou korunu. Možnost volby této hlavní měny by vyžadovala změny v logice aplikace i databázi, ale byla by realizovatelná v rámci rozumných mezí. Tento krok by dával smysl při překladu aplikace.

Závěr

Vyvinutá multiplatformní aplikace Budget Buddy umožňuje jednoduchou správu osobních financí. Snažil jsem se o vylepšení nedostatků existujících aplikací a zjednodušení celého procesu pro uživatele. Multiplatformnost dává uživateli možnost zvolit si libovolné zařízení pro užívání. Použití systému Material Design jinde než na domovské platformně Android považuji za korektní, jelikož aplikace od společnosti Google jsou velmi rozšířené a tento systém užívají na různých platformách.

Pro nasazení aplikace do produkce by bylo vhodné umožnit synchronizaci dat mezi zařízeními skrz server. Tím by se vyřešil problém s přístupem k datům na různých zařízeních. Tento krok by vyžadoval i zavedení uživatelských účtů a přihlašování.

Conclusions

The developed cross-platform application, Budget Buddy, allows for easy management of personal finances. I aimed to improve the shortcomings of existing applications and simplify the entire process for the user. Cross-platform support gives the user the ability to choose their preferred device for use. The use of the Material Design system on platforms other than the home Android platform is considered appropriate, as Google's applications are widely used across different platforms and utilize this system.

For deployment of the application to production, it would be useful to enable data synchronization between devices through a server. This would solve the issue of accessing data across different devices. This step would also require the introduction of user accounts and login functionality.

A Obsah elektronických dat

Součástí textu práce jsou i elektronická data v systému Katedry informatiky PřF v Olomouci s touto strukturou:

`src/`
: Adresář obsahující zdrojový kód aplikace. Jde o celý Flutter projekt.

`bin/`
: Adresář obsahující soubory pro instalaci na platformy Android a Windows.

`text/`
: Adresář s textem práce ve formátu PDF, vytvořený s použitím závazného stylu KI PřF UP v Olomouci pro závěrečné práce, včetně všech (textových) příloh, a všechny soubory potřebné pro bezproblémové vytvoření PDF dokumentu textu, tj. zdrojový text textu a příloh, vložené obrázky, apod.

`README.txt`
: Textový soubor s instrukcemi pro instalaci a spuštění aplikace.

Literatura

[1] *Cashew*. [online]. [cit. 2025-4-5]. Dostupný z: ⟨https://cashewapp.web.app/⟩.

[2] *What is wallet - Wallet by BudgetBakers - Your New Personal Finance Manager*. [online]. [cit. 2025-4-5]. Dostupný z: ⟨https://budgetbakers.com/what-is-wallet/⟩.

[3] *MoneyManager Ex*. [online]. [cit. 2025-4-5]. Dostupný z: ⟨https://moneymanagerex.org/⟩.

[4] *HomeBank | Free personal finance software, money management for everyone*. [online]. [cit. 2025-4-5]. Dostupný z: ⟨https://www.gethomebank.org/en/index.php⟩.

[5] *The Six Most Popular Cross-Platform App Development Frameworks | Kotlin Multiplatform Development Documentation*. [online]. [cit. 2025-4-5]. Dostupný z: ⟨https://www.jetbrains.com/help/kotlin-multiplatform-dev/cross-platform-frameworks.html⟩.

[6] *Dart overview | Dart*. [online]. [cit. 2025-4-5]. Dostupný z: ⟨https://dart.dev/overview⟩.

[7] *UI | Flutter*. [online]. [cit. 2025-3-29]. Dostupný z: ⟨https://docs.flutter.dev/ui⟩.

[8] *Showcase - Flutter apps in production*. [online]. [cit. 2025-4-5]. Dostupný z: ⟨https://flutter.dev/showcase⟩.

[9] *Human Interface Guidelines | Apple Developer Documentation*. [online]. [cit. 2025-5-1]. Dostupný z: ⟨https://developer.apple.com/design/human-interface-guidelines⟩.

[10] *Material Design*. [online]. [cit. 2025-3-29]. Dostupný z: ⟨https://m3.material.io/⟩.

[11] *SQLite Home Page*. [online]. [cit. 2025-3-29]. Dostupný z: ⟨https://www.sqlite.org/⟩.

[12] *Home - Drift*. [online]. [cit. 2025-4-5]. Dostupný z: ⟨https://drift.simonbinder.eu/⟩.

[13] *go-router | Flutter package*. [online]. [cit. 2025-4-8]. Dostupný z: ⟨https://pub.dev/packages/go_router⟩.

[14] *skeletonizer | Flutter package*. [online]. [cit. 2025-4-8]. Dostupný z: ⟨https://pub.dev/packages/skeletonizer⟩.

[15] Tankala, Samitha. *Skeleton Screens 101* [online]. [cit. 2025-4-6]. Dostupný z: ⟨https://www.nngroup.com/articles/skeleton-screens/⟩.

[16] *another-flushbar | Flutter package*. [online]. [cit. 2025-4-8]. Dostupný z: ⟨https://pub.dev/packages/another_flushbar⟩.

[17] *Components – Material Design 3*. [online]. [cit. 2025-4-5]. Dostupný z: ⟨https://m3.material.io/components⟩.

[18] *syncfusion_flutter_charts | Flutter package*. [online]. [cit. 2025-4-8]. Dostupný z: ⟨https://pub.dev/packages/syncfusion_flutter_charts⟩.

[19] *file_picker | Flutter package*. [online]. [cit. 2025-4-8]. Dostupný z: ⟨https://pub.dev/packages/file_picker⟩.

[20] *Bloc State Management Library | Bloc*. [online]. [cit. 2025-4-18]. Dostupný z: ⟨https://bloclibrary.dev/⟩.

[21] *Model-View-ViewModel - .NET | Microsoft Learn*. [online]. [cit. 2025-4-7]. Dostupný z: ⟨https://learn.microsoft.com/en-us/dotnet/architecture/maui/mvvm⟩.

[22] *Docs | Flutter*. [online]. [cit. 2025-4-8]. Dostupný z: ⟨https://docs.flutter.dev/⟩.

[23] *ISO - ISO 4217 — Currency codes*. [online]. [cit. 2025-4-9]. Dostupný z: ⟨https://www.iso.org/iso-4217-currency-codes.html⟩.

[24] *Kurzy devizového trhu - Česká národní banka*. [online]. [cit. 2025-4-9]. Dostupný z: ⟨https://www.cnb.cz/cs/financni-trhy/devizovy-trh/kurzy-devizoveho-trhu/kurzy-devizoveho-trhu/⟩.