STREDNÁ PRIEMYSELNÁ ŠKOLA ELEKTROTECHNICKÁ
KOMENSKÉHO 44, 040 01 KOŠICE

MOTORMINDER

PETER SZEPESI

2025

STREDNÁ PRIEMYSELNÁ ŠKOLA ELEKTROTECHNICKÁ
KOMENSKÉHO 44, 040 01 KOŠICE

MOTORMINDER

Záverečná práca

Praktická časť odbornej zložky maturitnej skúšky
Forma: Obhajoba vlastného projektu

2025
Košice

Riešitelia:
Peter SZEPESI

Ročník štúdia: štvrtý

Konzultant:
Ing. Miroslav BARUTIAK

Obsah

0 Úvod

Na začiatku nášho projektu sme nadobudli inšpiráciu počas letných prázdnin. Počas cesty autom domov z brigády, sme si všimli, že človek najazdí veľa kilometrov pri cestovaní do a z práce autom. Táto myšlienka nás viedla vytvoriť mobilnú aplikáciu, ktorá by nám umožnila zapisovať si kilometre, výdavky na pohonné hmoty ale taktiež nastaviť si upozornenia na prípadné servisy alebo kontroly o našich autách.

Náš projekt pozostáva z dvoch častí. Aplikácia pre mobilné zariadenia a lokálna databáza pre prihlasovanie používateľa a pre uschovanie zapisovaných dát. Rozhodli sme sa vyvinúť našu aplikáciu pre mobilné zariadenia s operačným systémom Android, pretože sme si uvedomili, že veľká časť ľudí používa práve tieto zariadenia. Aplikácia bola naprogramovaná v programovacom jazyku Java vo vývojom prostredí Android Studio. Android Studio ponúka plnú podporu pre jazyk Java, čo bola hlavnou motiváciou pri výbere prostredia. Počas vývoja našej aplikácie sme sa zamerali aj na vytvorenie a správu lokálnej databázy, ktorá zohráva kľúčovú rolu pri ukladaní a spracovaní dát od používateľov. Pre túto úlohu sme sa rozhodli použiť SQLite, ktorý je ľahký a jednoduchý relačný databázový systém vhodný pre mobilné aplikácie.

Na začiatku sme riešili meno a logo našej aplikácie a taktiež dizajn a farebnú schému aplikácie. Pri vytváraní dizajnu sme využili jazyk XML, čo je štandardný jazyk pri vytváraní užívateľského rozhrania v Android. Pri vytváraní loga sme použili internetovú stránku Canva a taktiež inšpiráciu od ostatných aplikácií, ktoré majú rovnaký zámer ako naša. Potom sme sa presunuli k vytvoreniu databázy a pripojeniu aplikácie na databázu. Pri tomto bode sme narazili na drobné problémy pri aktualizácií databázy, všetko sa však podarilo vyriešiť a mohli sme pokračovať ku funkcionalite aplikácie. Ako posledný krok sme napísali dokumentáciu a vytvorili prezentáciu.

Chceli by sme sa poďakovať pánovi inžinierovi a triednemu, Miroslavovi Barutiakovi, ktorý nám poskytoval mentálnu podporu pre náš projekt a inšpiroval nás skvelými nápadmi ako vylepšiť náš projekt.

1 Problematika a prehľad literatúry

Po skončení vývoja aplikácie budeme môcť v práci nájsť využité vedomosti a zručnosti získané za účelom vytvorenia projektu. Tieto vedomosti zahŕňajú: základy programovacieho jazyka Java, ich rozšírenie v rámci vývojového prostredia Android Studio, implementovanie a práca s relačnou databázou SQLite. Pri dizajne sme použili jazyk XML, ktorý je štandardným jazykom pri tvorbe používateľského rozhrania.

1.1 Prehľad existujúcich riešení

Aj keď existuje niekoľko aplikácií zameraných na sledovanie údajov o autách, avšak v niektorých z nich chýbajú alebo sú spoplatnené funkcie s ktorými naša aplikácia disponuje. Napríklad podpora viacerých vozidiel je často spoplatnená alebo nie je dostupná vo všetkých aplikáciách. Naša aplikácia však umožňuje používateľom sledovať si údaje o viacerých vozidlách bezplatne, čo je výhodou pre majiteľov viac vozidiel alebo rodiny s viacerými vozidlami. Taktiež nie všetky aplikácie ponúkajú notifikačné funkcie na pravidelné servisné kontroly a údržbové práce. Okrem toho naša aplikácia je viacjazyčná. Je k dispozícii v jazykoch: anglický, slovenský a maďarský.

1.2 Použité technológie

1.2.1 Android Studio

Pre vývoj našej aplikácie sme využili vývojové prostredie Android Studio. Je to integrované prostredie (IDE) špecificky pre systém Android. Android Studio ponúka prepracované prostredie pre vývoj aplikácií pre operačný systém Android a je široko využívaným. Taktiež ponúka plnú podporu pre jazyk Java, ktorý sme použili pri programovaní aplikácie, a taktiež sme mali prístup k množstvu nástrojov a knižníc, ktoré nám uľahčili proces vývoja.

Výhody:

- plná podpora pre vývoj aplikácií
- integrované prostredie pre vývoj
- široká podpora a komunita
- pravidelná aktualizácia a podpora[1]

Nevýhody:

- vysoké hardvérové nároky
- závislosť od internetu
- ťažké začiatky pre nováčikov[1]

1.2.2 Java

Jeden z najpopulárnejších a najvyužívanejších programovacích jazykov, ktorý je masovo používaný od jeho začiatkov až dodnes[2].

Bol vyvinutý spoločnosťou Oracle. Najdôležitejšou vlastnosťou tohto jazyka je objektová orientácia. Objektovo orientované jazyky sú také, kde obsah programu je napísaný pomocou objektov(protiklad objektovo orientovaného jazyka je procedurálny jazyk, kde obsah programu je napísaný v statických funkciách napr. jazyk C). Dátový typ objektu je trieda. Každý objekt má svoje vlastnosti a metódy, pomocou ktorých je objekt schopný vykonávať rôzne činnosti. Ďalšou dôležitou vlastnosťou je platformová nezávislosť. Zdrojový kód sa skompiluje do takzvaného „byte-kódu“ , ten sa na danom operačnom systéme interpretuje pomocou interpretera (Java Virtual Machine)[3].

1.2.3 XML

XML (eXtensible Markup Language) je značkovací jazyk. Tento jazyk je používaný na popis dát. Je odvodený od štandardného značkovacieho jazyka SGML (Standard Generalized Markup Language)[4]. V Android aplikáciách sa XML používa na implementáciu údajov týkajúcich sa používateľského rozhrania. XML obsahuje iba značky, pri implementácií stačí len ich volať.

Pri vytváraní rozhrania pre Android aplikáciu, XML súbory umožňujú oddeliť dizajn aplikácie od kódu, ktorý riadi jej logiku. Okrem definície používateľského rozhrania sa XML v Android používa aj pre definíciu zdrojov, ako sú reťazce, farby a dimenzie, čo pomáha pri lokalizácií a dostupnosti pre rôzne jazyky a zariadenia

1.2.4 SQLite

SQLite je relačný databázový systém, ktorý je ideálne vhodný pre vývoj mobilných aplikácií. Jeho použitie je nie len obmedzené na Android, používa sa aj v desktopových a webových aplikáciách. Jeho hlavné prednosti spočívajú v kompaktnosti a spoľahlivosti. Keďže SQLite používa štandardný jazyk SQL, umožňuje vývojárom pohodlne pracovať s databázovými operáciami ako sú vytváranie, čítanie, aktualizácia a mazanie dát s použitím bežných SQL príkazov.[5]

Výhody:

- **Nízka náročnosť na systémové zdroje:** SQLite nevyžaduje veľké množstvo pamäte ci procesorového výkonu
- **Flexibilita:** Jeho schopnosť ukladať dáta do jediného súboru uľahčuje distribúciu a zálohovanie
- **Rozsiahla podpora:** SQLite je široko podporovaný a integrovateľný s množstvom programovacích jazykov a platformami[6]

Nevýhody:

- **Obmedzené funkcionality:** Hoci SQLite podporuje veľkú časť štandardného SQL, môžu mu chýbať niektoré pokročilé funkcionality väčších databázových systémov
- **Nie je optimalizovaný pre veľké aplikácie:** Pre aplikácie s väčším objemom dát môže byť SQLite menej vhodné[6]

1.3 Finančná analýza

Ceny existujúcich aplikácií sa môžu líšiť v závislosti od ich funkčnosti a ponúkaných možností. Niektoré aplikácie môžu byť zadarmo s možnosťou nákupu rozšírených funkcií prostredníctvom mikrotransakcií, zatiaľ čo iné môžu byť platené s jednorazovým poplatkom na ich použitie. Aj keď sme zatiaľ neinvestovali do nášho projektu, avšak v prípade uvedenia aplikácie na Google Play Store si vyžaduje jednorazový poplatok vo výške 25$ (čo je približne 23€)[7]. Tento poplatok pokrýva registráciu účtu pre vývojárov a umožňuje nám uverejniť a priamo aktualizovať náš projekt na tejto platforme. Keby sme chceli zverejniť našu aplikáciu

na túto platformu, tak by sme ju zverejnili bez poplatkov. Tento model je veľmi obľúbený medzi vývojármi, pretože umožňuje rýchlejšie dosiahnuť širokú skupinu používateľov.

Celkové náklady na vývoj našej aplikácie

Technológia	**Cena**	**Hodiny**
XML	0€	15+
JAVA	0€	70+
SQLite	0€	35+

2 Výsledky práce

Naša aplikácia bola vytvorená tak, aby bola vhodná pre jednoduché používanie kýmkoľvek, kto by chcel mať väčší prehľad o svojich vozidlách. Aplikáciu sa dá vyexportovať s príponou .apk. Takže ju je možné používať a využívať na mobilných zariadeniach s operačným systémom Android.

Po nainštalovaní a spustení sa nám objaví úvodná stránka pre prihlásenie sa. Ak používateľ nemá účet, tak sa môže zaregistrovať pomocou tlačidla na obrazovke. Po registrácií a prihlásení je používateľ prihlásený aj po vypnutí aplikácie, ak si zvoli možnosť zapamätať si ma.

2.1 Databáza aplikácie

Náš projekt obsahuje databázu na skladovanie dát, ktorá je dôležitou časťou projektu. Databázu nastavujeme v triede „DatabaseHelper“, kde sa nachádza definícia databázy a taktiež metódy na manipuláciu sa dátami. Trieda „DatabaseHelper“ dedí metódy z triedy „SQLiteOpenHelper“. Táto trieda obsahuje potrebné metódy na vytvorenie a manažovanie databázy ako sú: „onCreate()“, „onUpgrade()“ a „onOpen()“[8]. Rozhodli sme sa písať metódy v tejto triede pre efektivitu a opätovné využitie kódu. Týmto spôsobom môžeme napísať metódy raz a potom použiť ich viackrát v celej aplikácií, čo zjednodušuje a urýchľuje proces vývoja. Taktiež nám to umožňuje udržiavať kód prehľadný a ľahko udržateľný, keďže všetky metódy sú sústredené na jednom mieste a nemusíme sa opakovať v iných častiach kódu. Databáza aplikácie sa volá Database.db.

Skladá sa zo siedmich tabuliek:

- „users_table”: Tabuľka do ktorej ukladáme informácie o používateľovi, ako sú ich meno, priezvisko, používateľské meno a heslo.
- „cars_table”: Tabuľka slúžiaca na uchovávanie informácií o vozidlách, ako sú meno auta, značka, model, rok výroby, počet najazdených kilometrov na tachometri, typ paliva, fotka o aute a taktiež poznámky o aute.
- „refuel_table”: Tabuľka pre záznamy o tankovaní paliva, ako sú dátum, čas, miesto tankovania, množstvo(v litroch, kWh alebo kg, podľa typu paliva) a cenu. Konečná cena za tankovanie sa vypočíta automaticky po zadaní množstva a cenu za jednotku paliva.
- „odometer_table“: Tabuľka ukladajúca informácie o prejdených kilometrov. Po pridaní alebo odstránení záznamu sa aktualizuje počet najazdených kilometrov v tabuľke

„cars_table“. Tabuľka obsahuje informácie ako sú dátum a počet najazdených kilometrov za ten deň.

- „expenses_table“: Tabuľka obsahujúce náklady spojené s prevádzkou vozidiel alebo len bežným opotrebovaním. Obsahuje informácie ako sú dátum, čas, miesto výdavku, cena opravy a typ výdavku.
- „services_table“: Tabuľka pre záznamy o servisoch vozidla. Obsahuje informácie ako sú dátum a čas opravy, miesto opravy, cena opravy a typ opravy.
- „notifications_table“: Tabuľka, ktorá obsahuje informácie o upozorneniach, ktoré si sám používateľ nastavil. V tabuľke sa nachádzajú informácie ako sú popis upozornenia, typ opravy, miesto, dátum kedy sa má upozornenie spustiť alebo v akých časových intervaloch sa má upozornenie opakovať.

Počas vývoju našej aplikácie sme narazili na problém spojený s aktualizáciou databázy. Zmeny, ktoré sme vykonali v štruktúre tabuliek alebo doménach, sa neprejavili v existujúcej databáze používateľa. Po preskúmaní problému sme zistili, že koreňom problému bola nemeniaca verzia databázy. V SQLite je nevyhnutné zvýšiť verziu databázy, aby systém vedel, že sa má vykonať aktualizácia štruktúry databázy pomocou metódy „onUpgrade()“ v triede „Databasehelper“. Riešenie tohto problému spočívalo v jednoduchej, ale dôležitej zmene: inkrementácii čísla verzie databázy v kóde, kde sme definovali databázu. Táto zmena signalizovala SQLite engine, aby spustil aktualizáciu databázy [9].

2.2 MainActivity a pridávanie auta

Vytváranie aplikácií sa vždy začína súborom MainActivity a rozložením aplikácie. Rozhodli sme sa nepoužiť vstavané rozloženie od Android Studio, ale všetky rozloženia si vytvoriť sami.

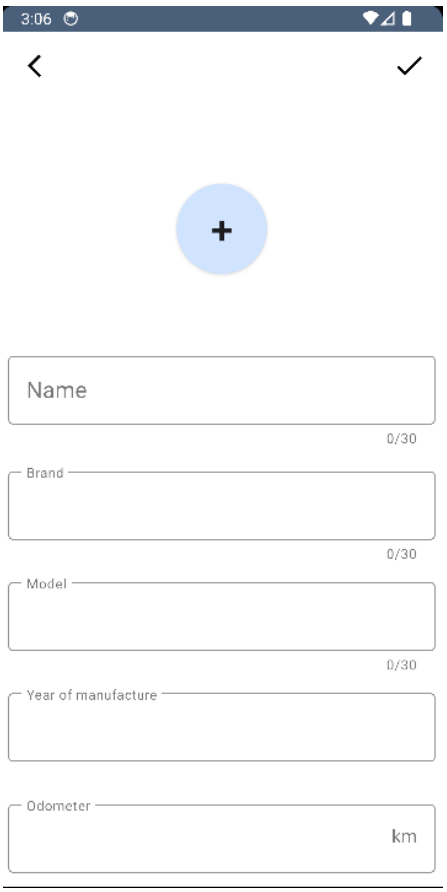
Hlavná aktivita je jedna z troch dôležitých aktivít tejto aplikácie. Ostatné doležíte aktivity sú: Štatistická aktivita a Notifikačná aktivita. V hlavnej aktivite vidíme najprv obrazovku bez pridaného auta. Užívateľ vidí základné rozhranie aplikácie, kde sa nachádzajú rôzne možnosti a funkcie, vrátane tlačidla „plus“. Toto tlačidlo slúži k pridaniu nového vozidla do aplikácie. Po stlačení tohto tlačidla, sa spusti udalosť, ktorá otvorí novú aktivitu s názvom „AddPage“. V tejto aktivite ma používateľ možnosť pridať si vozidlo do aplikácie.

V „AddPage“ sa nachádza formulár. Povinné sú informácie ako značka auta, model auta, rok výroby a počet najazdených kilometrov. Používateľ taktiež môže zadať „meno“ auta, ktoré pomôže pri vyberaní auta z výpisu zadaných aut. Taktiež si používateľ môže vybrať fotku

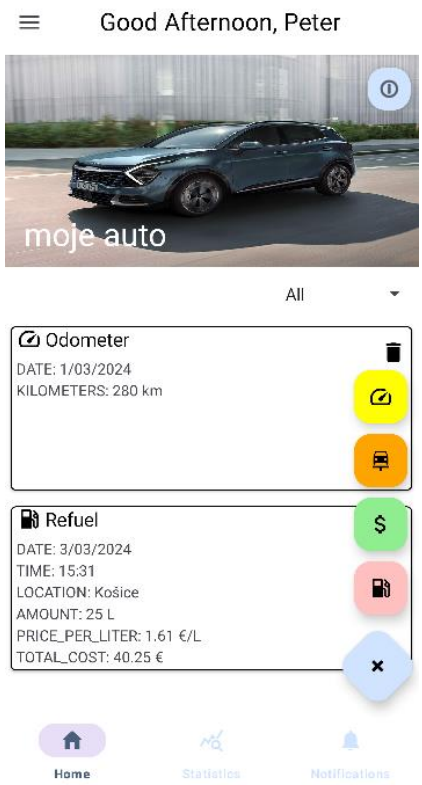
o aute z galérie zariadenia a taktiež si napísať dajaké poznámky o aute v políčku „Notes“. Po stlačení tohto tlačidla „pridať“ sú zadané informácie uložené do databázy a zobrazenie sa prepne naspäť na hlavnú aktivitu. Používateľ teraz vidí svoje pridané auto v hlavnej aktivite.

Po pridaní auta a opätovnom stlačení tlačidla „plus“, sa otvorí menšie menu pozostávajúce zo štyroch tlačidiel. Každé tlačidlo v tomto menu reprezentuje konkrétnu kategóriu udalosti a po stlačení sa otvorí príslušná aktivita, kde používateľ môže vložiť podrobnosti o danej udalosti. Napríklad, po stlačení tlačidla pre zápis o prejdených kilometroch sa otvorí aktivita, kde používateľ si môže zadať dátum a počet prejdených kilometrov vozidlom. Podobne, ostatné tlačidlá umožňujú požívateľovi vytvoriť si záznamy o servise, výdavkoch a tankovaní paliva súvisiacich s daným vozidlom. Po pridaní záznamu sa obrazovka vráti na hlavnú aktivitu a vytvorí sa kartička s daným záznamom, ktorý sa dá vymazať spolu s celým záznamom.

Obr. 2.2.-1



Obr. 2.2.-2



Obr. 2.2.-3

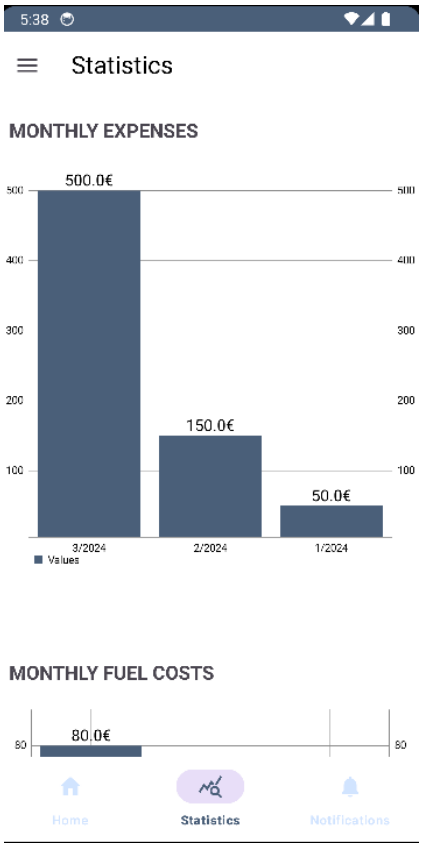
```
// Získanie dát z textových polí
String carNameStr = carNameEditText.getText().toString();
String carBrandStr = carBrandEditText.getText().toString();
String carModelStr = carModelEditText.getText().toString();
String carYearStr = carYearEditText.getText().toString();
String carOdometerStr = carOdometerEditText.getText().toString();
String selectedFuelType = fuelTypeSpinner.getSelectedItem().toString();

// Premenná pre počítanie chýb pri validácii údajov
int errorCount = 0;

// Validácia dát pomocou triedy Error
errorCount += Error.getError(
        new TextInputEditText[]{carBrandEditText, carModelEditText, carYearEditText, carOdometerEditText},
        new TextInputLayout[]{brandInputLayout, modelInputLayout, yearInputLayout, odometernputLayout},
         context: AddPage.this,
        "This field cannot be empty!"
);

// Validácia roku výroby
try {
    int carYear = Integer.parseInt(carYearStr);
    if (carYear < lowestYear || carYear > getCurrentYear()) {
        yearInputLayout.setError("Year must be between 1900 and" + getCurrentYear());
        errorCount++;
    }
} catch (Exception e) {
}

// Validácia stavu odometra
try {
    float carOdometer = Float.parseFloat(carOdometerStr);
    if (carOdometer < lowestOdometer || carOdometer > highestOdometer) {
        odometernputLayout.setError("Odometer must be between 0 and 1,000,000");
        errorCount++;
    }
} catch (Exception e) {
```

Obr. 2.2.-4

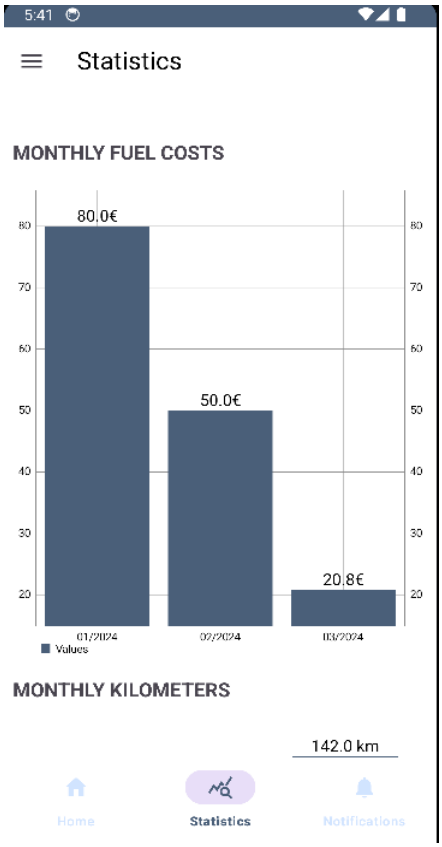
2.3 StatisticsActivivy

Druhou hlavnou aktivitou je „StatisticsActivity“. V tejto aktivite sa nachádzajú tri grafy na zobrazenie rôznych údajov. Tieto grafy vieme vytvoriť pomocou komponenty „BarChart“[10]. Bar chart je typ grafu, ktorý vizualizuje dáta pomocou stĺpcov. Každý stĺpec predstavuje určitú hodnotu a jeho výška je priamo úmerná tejto hodnote. Hodnoty v grafe sú usporiadané podľa mesiaca a roku. Údaje pre prvý graf sú vybrané z databázy pomocou metódy „displayMonthlyExpensesandServices“. Pre druhý a tretí graf sa využíva metóda „createBarChart“, ktorá vyberá údaje z tej tabuľky, ktorú sme zadali do parametrov tejto funkcie. V prvom grafe sú zobrazené spojené peňažné údaje na výdavky a servisné práce v tom mesiaci. V druhom grafe sú zobrazené peňažné údaje na výdavky pri tankovaní nášho vozidla v tom mesiaci. V treťom a poslednom grafe sú zobrazené prejdené kilometre za príslušný mesiac

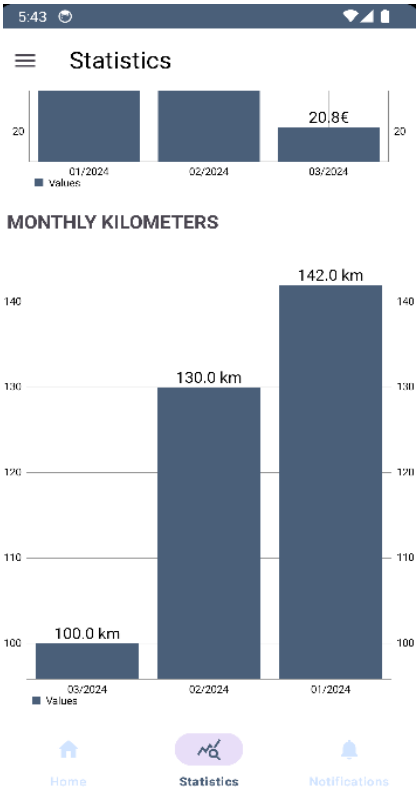
.



Obr. 2.3.-1



Obr. 2.3.-2



Obr. 2.3.-3

2.4 NotificationsActivity

Treťou a poslednou hlavnou aktivitou je „NotificationsActivity“. V tejto aktivite má používateľ možnosť nastaviť si upozornenia pre prípadné práce s autom alebo len všeobecnú pripomienku o aute. Po kliknutí na tlačidlo „plus“ sa zobrazí formulár kde používateľ môže zadať informácie o pripomienke a taktiež nastaviť si čas spustenia. Pri výbere času spustenia môže si používateľ vybrať medzi možnosťami: opakujúce sa upozornenie a upozornenie, ktoré sa spustí len raz. Pri výbere upozornenia, ktoré sa spustí raz je to celkom jednoduché. Používateľ si výbere dátum a upozornenie sa spustí počas toho dňa. Pri výbere upozornenia, ktoré sa má pravidelne opakovať, si používateľ vyberie v akom intervale sa má upozornenie opakovať. Intervaly sú od týždenného opakovania až po ročné opakovanie.

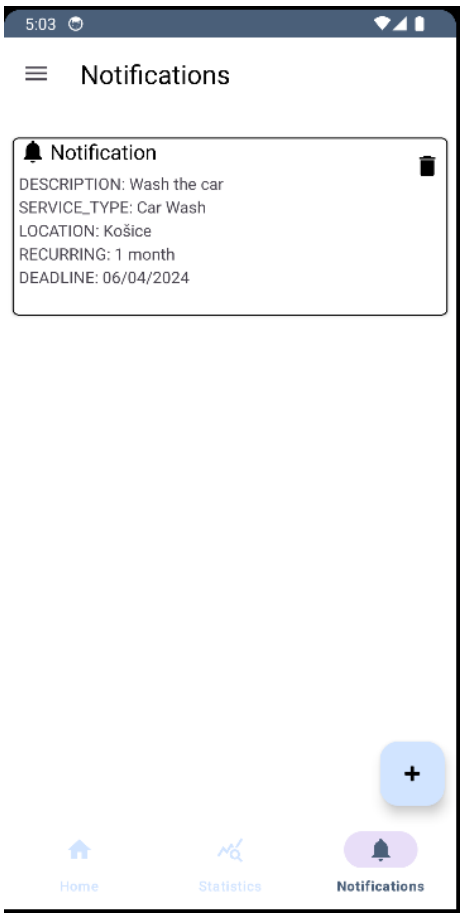
Po zadaní dátumu od používateľa sa čas do tohto dátumu prevedie na milisekundy. Tento údaj je potom použitý na naplánovanie upozornenia pomocou metódy „setAlarmClock“, ktorá sa nachádza v triede „AlarmManager“. „AlarmManager“ je trieda, ktorá poskytuje prístup k systémovým službám pre upozornenia[11]. Je používaná na naplánovanie či spustenie určitých úloh v budúcnosti v aplikácií. Keď čas v milisekundách uplynie, tak sa pošle PendingIntent, ktorý zavolá metódu „onRecieve“ v triede „BroadcastReciever“, čím sa vygeneruje upozornenie. „BroadcastReciever“ je trieda, ktorá umožňuje aplikácií reagovať na

udalosti. Je to trieda, ktorá prijíma intent zo systému alebo iných aplikácií a vykoná príslušnú akciu[12]. Obsahuje metódu „onReceive“, ktorá sa volá keď „BroadcastReceiver“ príjme vyslaný intent[13].

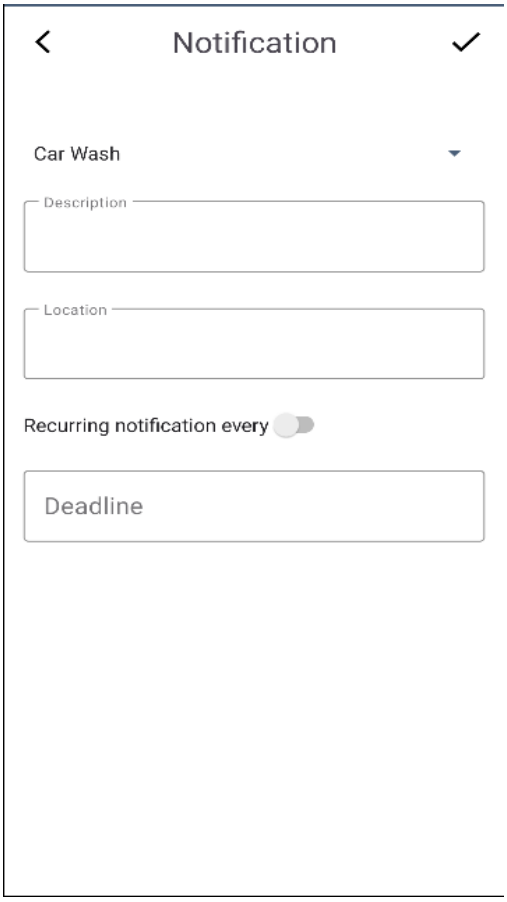
Po pridaní upozornenia sa vytvorí kartička, ktorá obsahuje všetky informácie o upozornení. Táto kartička sa samo aktualizuje dátum spustenia v prípade, že upozornenie sa má opakovať. V prípade jednorazového upozornenia sa samo vymaže.

```
    // Vytvorenie PendingIntent pre notifikáciu
    PendingIntent pendingIntent = PendingIntent.getBroadcast(
            context,
            notificationId,
            intent,
             flags: PendingIntent.FLAG_UPDATE_CURRENT | PendingIntent.FLAG_IMMUTABLE
    );


    // Kontrola, či aplikácia môže naplánovať presné alarmy
    if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.S) {
        AlarmManager.AlarmClockInfo alarmClockInfo = new AlarmManager.AlarmClockInfo(deadlineMillis, pendingIntent);
        try {
            // Pokus o nastavenie presného alarmu
            alarmManager.setAlarmClock(alarmClockInfo, pendingIntent);
            Log.d( tag: "NotificationTest",  msg: "Scheduled exact alarm");
        } catch (SecurityException e) {
            // výpis výnimky SecurityException
            Log.e( tag: "NotificationTest",  msg: "SecurityException: " + e.getMessage());
        }
    } else {
        // Pre staršie verzie Androidu použijeme setExact
        alarmManager.setExact(AlarmManager.RTC_WAKEUP, deadlineMillis, pendingIntent);
        Log.d( tag: "NotificationTest",  msg: "Scheduled non-exact alarm");
    }
} catch (Exception e) {
    //výpis erroru, keď zlyhá naplánovanie upozornenia
    e.printStackTrace();
}
```

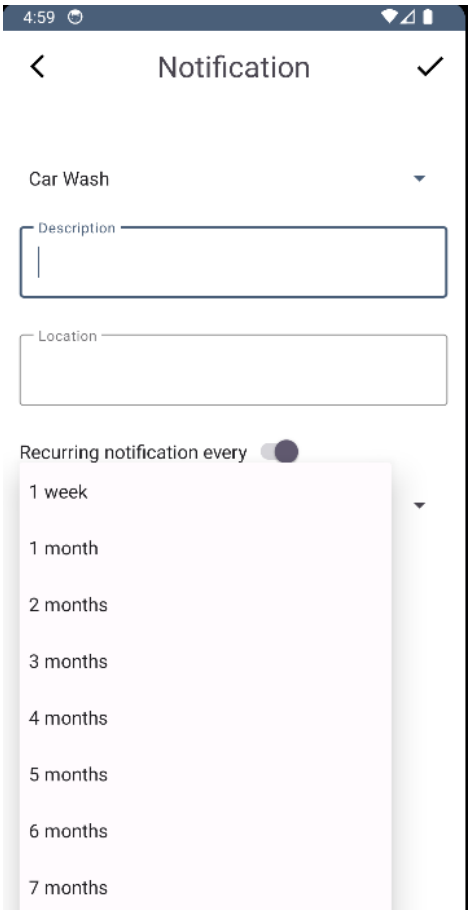
Obr. 2.4.-1



Obr. 2.4.-2



Obr. 2.4.-3



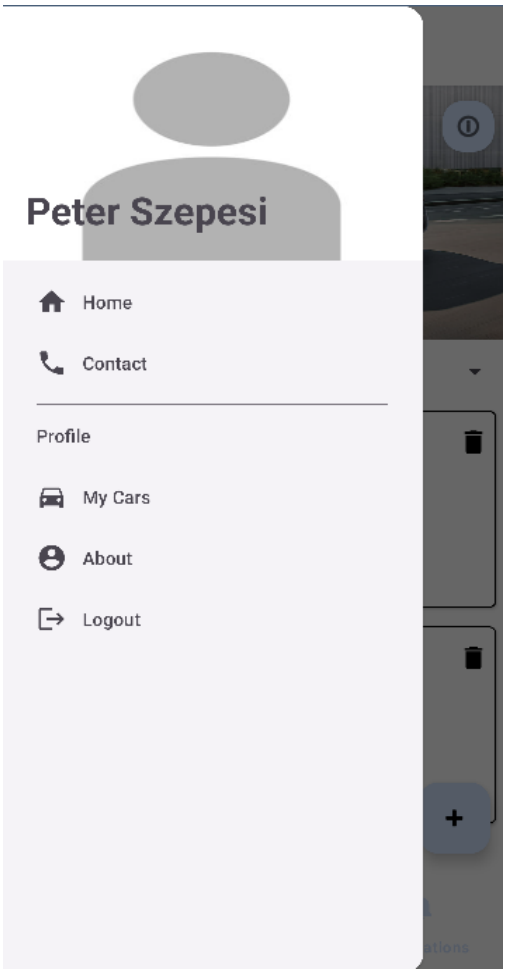
Obr. 2.4.-4

2.5 Bočné navigačné menu

V našej aplikácií sme implementovali bočné navigačné menu, ktoré poskytuje používateľom jednoduchý prístup k funkciám a sekciám aplikácie. Vytváranie bočného menu si vyžaduje použitie komponenty DrawerLayout. DrawerLayout je komponent, ktorý umožňuje vložiť hlavný obsah aplikácie a bočné navigačné menu do rovnakej obrazovky[14]. Na definovanie obsahu bočného menu sa používa NavigationView. Tento komponent umožňuje definovať jednotlivé položky menu a priradiť im ikony a popisy[15]. V kóde aplikácie sa potom implementuje logika, ktorá umožňuje otváranie a zatváranie menu. To zahŕňa reakciu na udalosti, ako je kliknutie na tlačidlo pre otvorenie menu alebo gesto ťahania prsta zľava do prava. Po výbere položky sa vykoná kód, ktorý reaguje na tento výber a vykoná otvorenie príslušnej aktivity.

Naše navigačné menu obsahuje tieto položky:

- Domov: Táto položka slúži na vrátenie používateľa na domovskú obrazovku (Main Activity). Táto položka sa často umiestňuje ako prvá do bočného menu, tak sme to spravili aj my.
- Kontakt: Táto položka umožňuje používateľovi získať kontaktné informácie o vývojároch aplikácie. Obsahuje informácie ako sú, telefóne číslo a emailovú adresu pre prípadné chyby v aplikácií.
- Zoznam pridaných áut: Asi najdôležitejšiu položkou v menu je zoznam pridaných áut. Táto aktivita umožňuje používateľovi pridávať, prehľadávať svoje vozidlá. Stačí len kliknúť na fotku vozidla a automaticky sa vám nastaví ako predvolené vozidlo.
- Profil: Používateľ ma prístup k informáciám o svojom profile. Táto aktivita zobrazuje informácie o používateľovi, ako sú meno, priezvisko a prihlasovacie meno. Taktiež umožňuje používateľovi odstrániť si účet po správnom zadaní hesla.
- Odhlásenie sa: Používateľ ma možnosť odhlásiť sa z aplikácie. Táto položka ukončí reláciu prihlásenia sa a vráti používateľa na obrazovku prihlasovania sa.



Obr. 2.5.-1

Obr. 2.5.-2

```java
// Metóda na spracovanie výberu položiek z bočného navigačného menu
1 usage
public void handleNavigationItemSelected(MenuItem menuItem) {
    switch (menuItem.getItemId()) {
        case R.id.home:
            NavigationUtils.openHome( context: this);
            break;
        case R.id.about:
            NavigationUtils.openAbout( context: this);
            break;
        case R.id.logout:
            logout();
            break;
        case R.id.contact:
            NavigationUtils.openContact( context: this);
            break;
        case R.id.cars:
            NavigationUtils.openMyCars( context: this);
            break;
        default:
            break;
    }
}
```

Obr. 2.5.-3

3 Závery a zhrnutie

Realizáciou nášho projektu sme priniesli riešenie pre ľudí, ktorých zaujíma sledovanie a správa údajov o ich vozidlách. Naším cieľom bolo zefektívniť tento proces zaznamenávania a zápis údajov ale taktiež aj nastavovanie si upozornení na údržbové práce na vozidle. Výhodou našej aplikácie je jednoduché prihlasovanie, intuitívne používateľské rozhranie, ktoré umožňuje používateľovi ľahko a efektívne zaznamenávať si údaje o vozidlách, čím prispeje k zlepšeniu celkovej užívateľskej skúsenosti.

Potenciál k zlepšeniu vidíme pri prihlasovaní používateľa, kde sme mohli využiť možnosť prihlasovania sa pomocou účtu Google. Táto možnosť prihlasovania by poskytla používateľom ďalšiu možnosť prihlasovania, čo by zjednodušilo proces registrácie a taktiež celkovú užívateľskú skúsenosť. Takáto funkcionalita by tiež prispela k zvýšeniu dôveryhodnosti našej aplikácie.

Problémov pri riešení projektu bolo viacero. Najväčšou výzvou s ktorou sme sa stretli, bola implementácia a správa databázy, čo si vyžiadalo dôkladne preskúmanie a ladenie. Problém vznikol pri aktualizácii verzie databázy . Taktiež boli problémy pri implementácii upozornení, kde sme sa stretli s rôznymi problémami. Dakedy upozornenia fungovali správne, inokedy však zlyhávali alebo sa spúšťali nekonečne. Napriek týmto prekážkam sme s vytrvalosťou dosiahli funkčnú aplikáciu.

Tento projekt nám poskytol cenné skúsenosti v oblasti tvorby Android aplikácií a prácu s databázou. Tieto skúsenosti nás posunuli vpred na vyučovacích hodinách, kde sa venujeme Android programovaniu a relačným databázam. Projekt nám otvoril cestu k prekonaniu technických výziev a rozšíreniu našich znalosti v oblasti vývoje aplikácií. Taktiež sme si rozšírili naše znalosti pri tvorbe dizajnu aplikácie pre mobilné zariadenia. Najväčšie plusy však vidíme v tom, že sme získali skúsenosti s návrhom a realizáciou projektu a taktiež riešením vzniknutých problémov. Veríme, že tieto skúsenosti budú mať pozitívny vplyv na našu budúcnosť, kde budeme môcť využiť naše nové poznatky.

4 Resumé

Our Android application serves as a tool for tracking and managing car-related data. Our objective was to streamline the process of recording kilometers, fuel expenses, service expenses and setting up maintenance related reminders for vehicles. Using the Java programming language and Android Studio, we created the application to ensure a user-friendly experience and a seamless functionality. We also paid attention to the user interface design, ensuring an appealing interface that gives the best user experience.

Our app also includes a SQLite database for user authentication and storing the recorded data locally on the device. We have had a few problems with this project. The biggest problems were setting up the database and the notifications, but after a few setbacks, we successfully implemented them.

Our app goes beyond basic functionality by offering support for multiple vehicles, customizable notifications and multi-language support. We hope these features will enhance the user experience and that our users will appreciate our app.

5 Zoznam použitej literatúry

5.1 Príklady bibliografických odkazov

5.1.1 Elektronické monografie, www stránky, databázy, programy:

[1] Pangea.ai. Pros and Cons of Android Studio and App Tools [online] Pangea 2022-11-03 [cit. 2024-03-10]. Dostupné na internete: <https://pangea.ai/blog/services/pros-and-cons-of-android-studio-and-app-tools>

[2] GLUSHACH, Roman. The Evolution of Java: A Historical Perspective [online]. Medium, 2023-08-11 [cit. 2024-03-10]. Dostupné na internete: <https://romanglushach.medium.com/the-evolution-of-java-a-historical-perspective-e15c3d7e5f85>.

[3] Oracle(2015). The Java Virtual Machine Specification Java SE 8 Edition [online] Oracle Corporation, 2015-02-13 [cit. 2024-03-10]. Dostupné na internete:

< https://docs.oracle.com/javase/specs/jvms/se8/html/jvms-1.html#jvms-1.2.>

[4] Wikipédia: Otvorená encyklopédia. XML[online]. Wikipédia 2017-05-13 [cit. 2024-03-10]. Dostupné na internete: <https://sk.wikipedia.org/wiki/XML>

[5] SQLite. About SQLite [online]. SQLite 2007-11-12. Aktualizované 2023-10-10 [cit. 2024-03-12]. Dostupné na internete: <https://www.sqlite.org/about.html>

[6]Christopher Wong. Advantages and Disadvantages of Using SQLite [online]. Medium 2023-03-31 [cit. 2024-03-13]. Dostupné na internete: <https://medium.com/@cw30355/advantages-and-disadvantages-of-using-sqlite-2f490fa467bd>

[7] Google. Požiadavky na aplikácie [online]. Google Play Console Help 2024 [cit. 2024-03-13].Dostupné na internete: <https://support.google.com/googleplay/android-developer/answer/6112435?hl=sk&sjid=15659249342562626661-EU>

[8] Google. SQLiteOpenHelper [online]. Android Developers. Aktualizované 2024-02-16 [cit. 2024-03-13]. Dostupné na internete:

<https://developer.android.com/reference/android/database/sqlite/SQLiteOpenHelper>

[9] MikeT. Android SQLite table column not being created or recognized. What am I missing? [online]. Stack Overflow 2019-04-19 [cit. 2024-03-13]. Dostupné na internete:

<https://stackoverflow.com/questions/55767442/android-sqlite-table-column-not-being-created-or-recognized-what-am-i-missing>

[10] Philipp Jahoda. MPAndroidChart [online]. WeeklyCoding 2021 [cit. 2024-03-13]. Dostupné na internete: < https://weeklycoding.com/mpandroidchart-documentation/>

[11] Google. AlarmManager [online]. Android Developers. Aktualizované 2023-06-07 [cit. 2024-03-14] Dostupné na internete:

<https://developer.android.com/reference/android/app/AlarmManager>

[12] Google. BroadcastReceiver [online]. Android Developers. Aktualizované 2023-06-07 [cit. 2024-03-14]. Dostupné na internete:

<https://developer.android.com/reference/android/content/BroadcastReceiver>

[13] Google. Broadcasts [online]. Android Developers. Aktualizované 2024-02-07 [cit. 2024-03-15]. Dostupné na internete: <https://developer.android.com/develop/background-work/background-tasks/broadcasts>

[14] Google. DrawerLayout [online]. Android Developers. Aktualizované 2023-11-02 [cit. 2024-03-16]. Dostupné na internete:

< https://developer.android.com/reference/androidx/drawerlayout/widget/DrawerLayout>

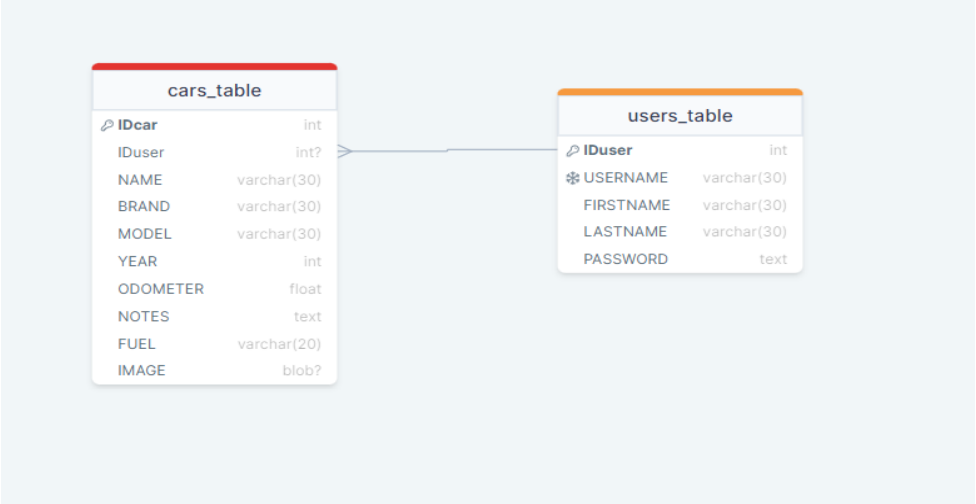
[15] Google. NavigationView [online]. Android Developers. Aktualizované 2023-12-15 [cit. 2024-03-16]. Dostupné na internete:

<https://developer.android.com/reference/com/google/android/material/navigation/NavigationView>

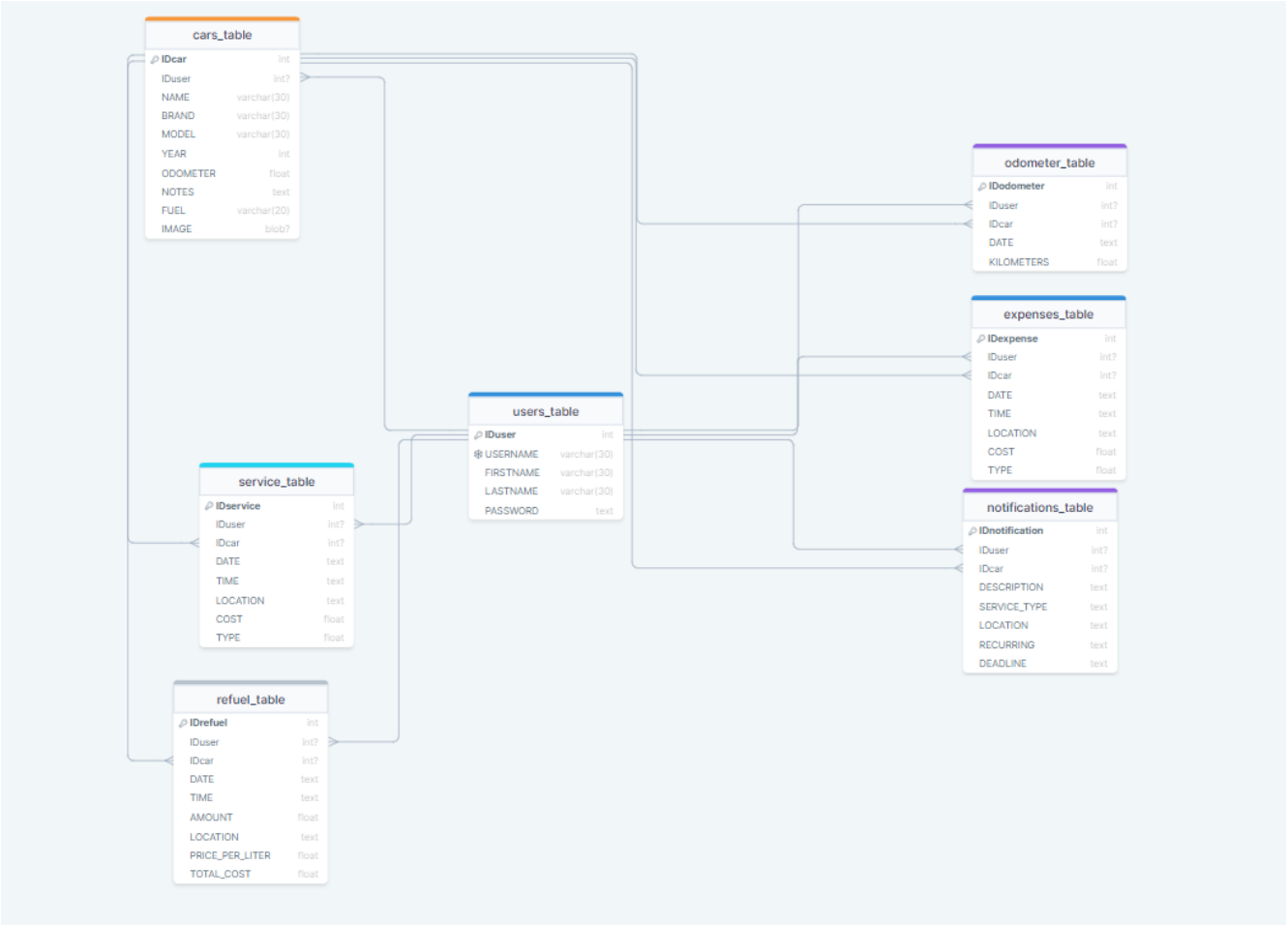
6 Prílohy

Príloha A – Databázové diagramy

Príloha A.1 – Entitno-relačný diagram medzi autom a používateľom

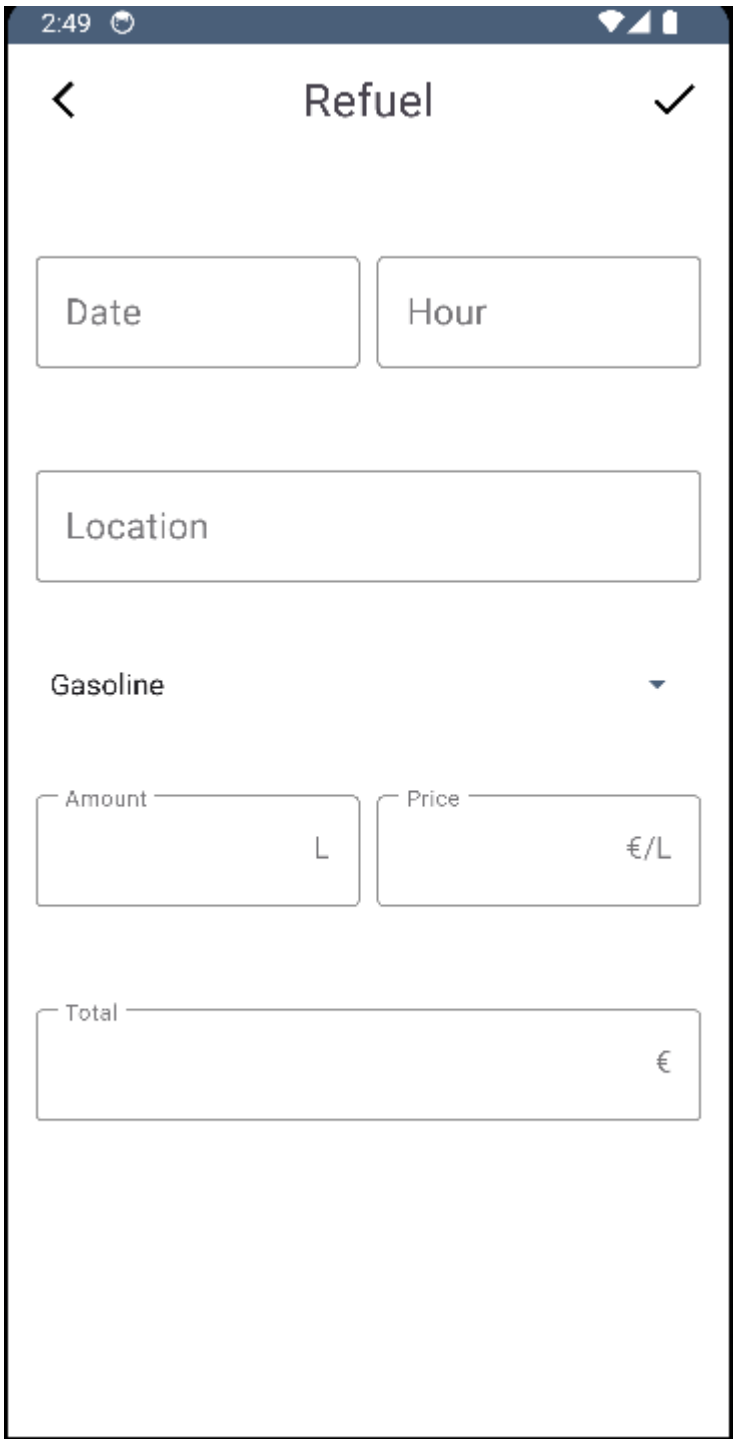


Príloha A.2 – Entitno-relačný diagram celej databázy

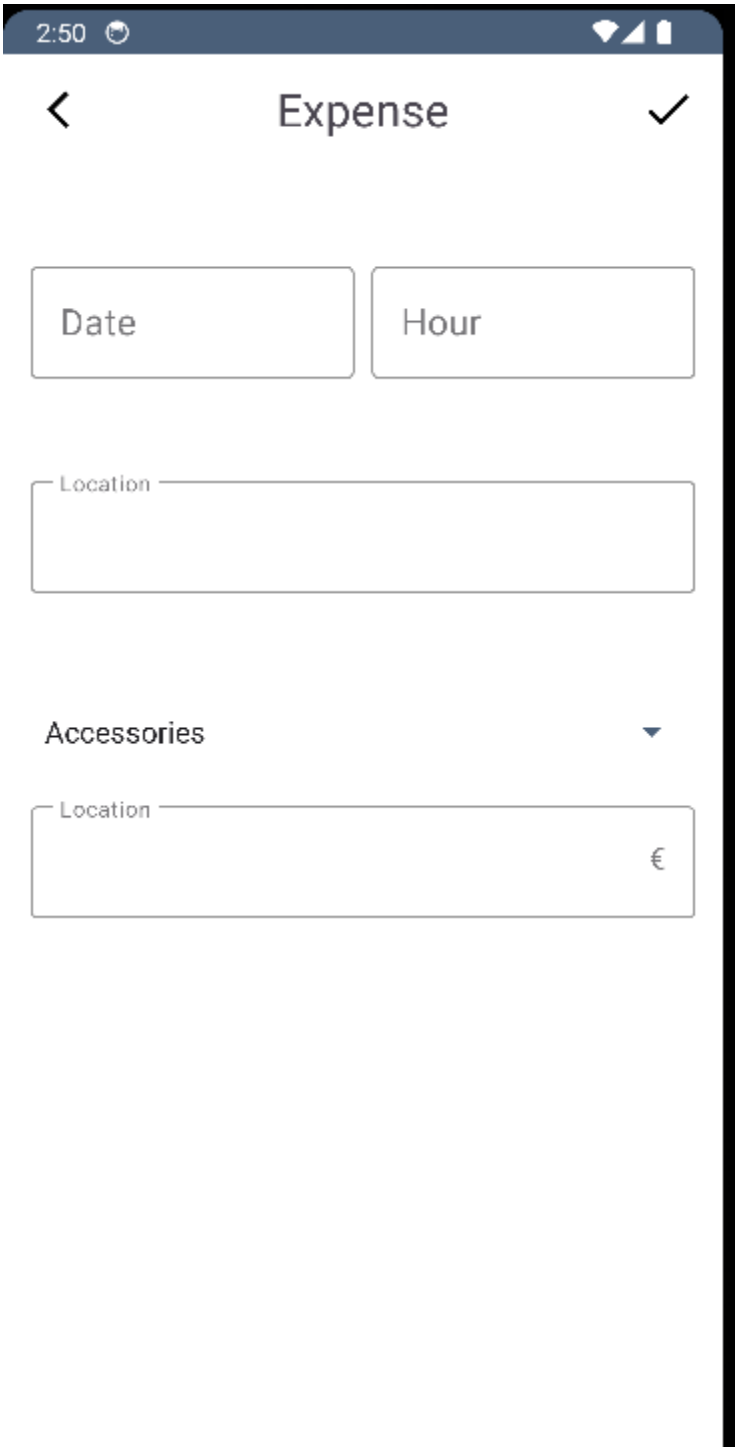


Príloha B – Zadávanie záznamov

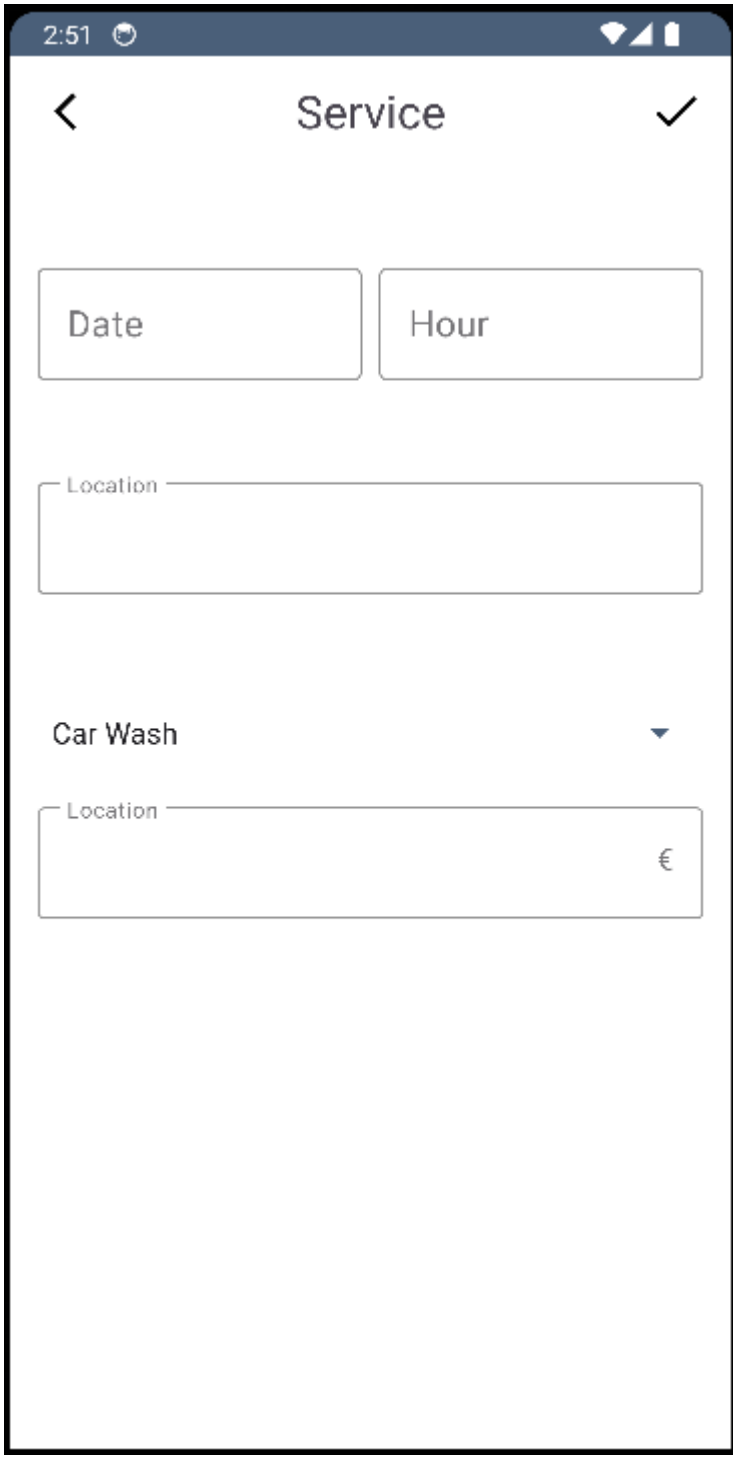
Príloha B.1 – Záznam pre tankovanie



Príloha B.2 – Záznam pre výdavok



Príloha B.3 – Záznam pre servis



Príloha B.4 – Záznam pre prejdené kilometre