ŽILINSKÁ UNIVERZITA V ŽILINE
Fakulta riadenia
a informatiky

Semestrálna práca z predmetu *Vývoj aplikácii pre mobilné zariadenia*

CAR-MAIN

Študijný program: Informatika
Vypracovali: Dominik Bubeník
Študijná skupina: 5ZYI21

v Žiline dňa 11. 6. 2024

1) Popis a analýza riešeného problému

1.1) Špecifikácia zadania

Aplikácia CAR-MAIN slúži na prehľadné zaznamenávanie údajov o používateľovom aute, prípadne autách. Ďalej aplikácia pomáha používateľovi sledovať približnú veľkosť výdavkov za vozidlo, či už v podobe tankovania, servisu, parkovania atď. Do aplikácie si vie používateľ zadať informácie o aute, napríklad značka vozidla, typ, motorizácia, druh paliva, rok výroby. Následne bude aplikácia uchovávať tieto informácie.

Majitelia vozidiel často nemajú prehľad o celkových nákladoch spojených s prevádzkou svojho vozidla. Výdavky na palivo, opravy, údržbu a iné povinné náklady sa môžu rýchlo hromadiť a spôsobovať finančné ťažkosti, ak nie sú správne sledované a plánované.

1.2) Prehľad podobných aplikácií na trhu

Na Google Play sa vyskytuje viacero podobných aplikácií. Líšia sa väčšinou iba grafickým spracovaním a drobnými funkcionalitami. Väčšina ponúka aj rozšírené možnosti používania, avšak za príplatok alebo pravidelný poplatok.

Simply Auto

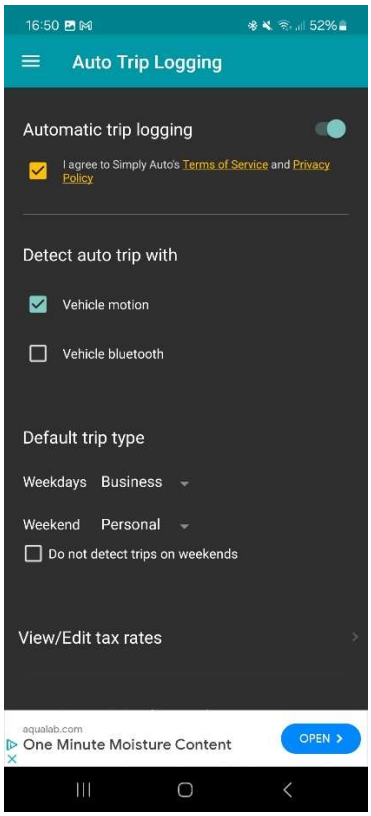
Simply Auto je typickým príkladom aplikácie, ktorá zaznamenáva údaje o vozidle a pomáha používateľovi neprešvihnúť dôležité dátumy.

Výhody:

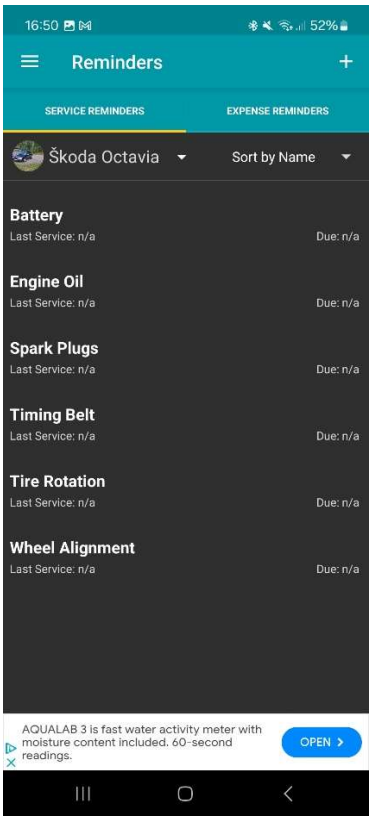
- prehľad údajov v tabuľkách
- pridávanie záznamov je pomerne intuitívne
- automatické zaznamenávanie jazdy
- používateľ si môže nastaviť pripomienky

Nevýhody:

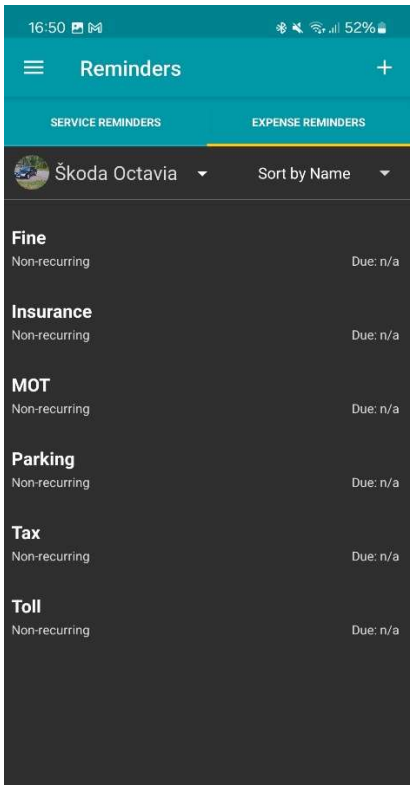
- Automatické zaznamenávanie nie je vždy presné (z recenzií používateľov)
- App neponúka celkové štatistiky v grafoch



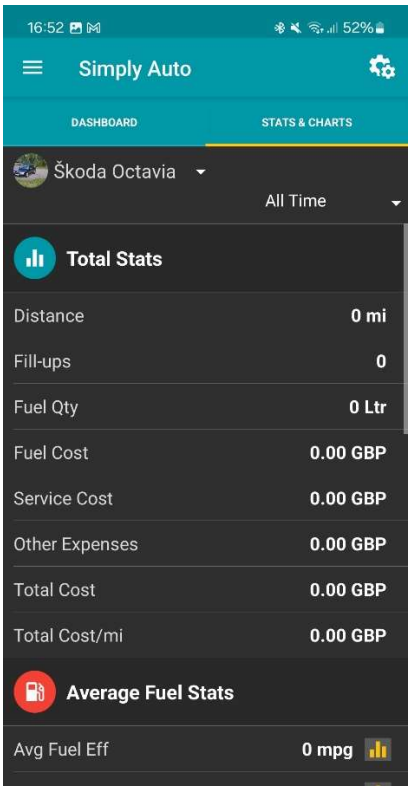
Automatický záznam jazdy



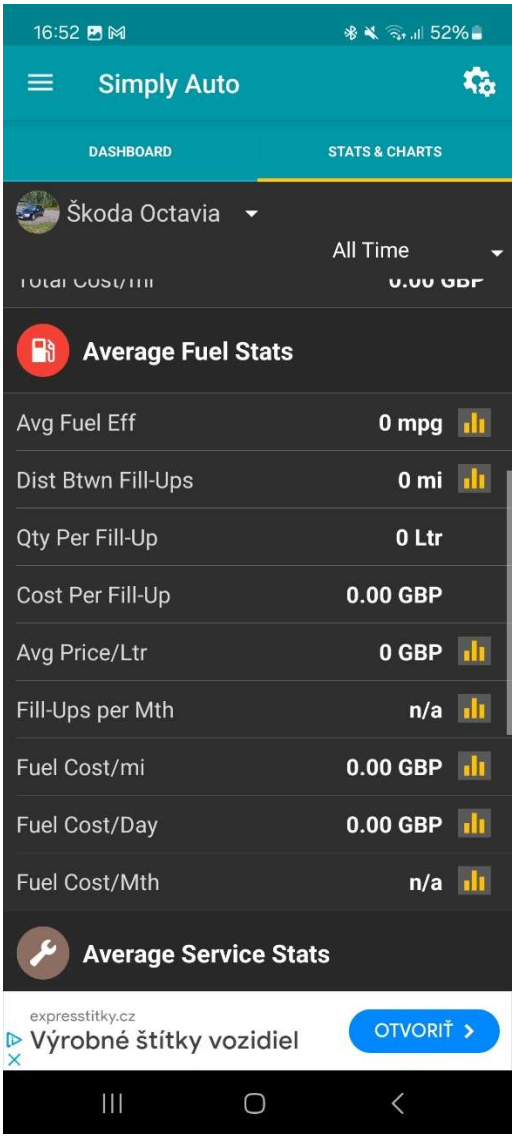
Servisné pripomienky

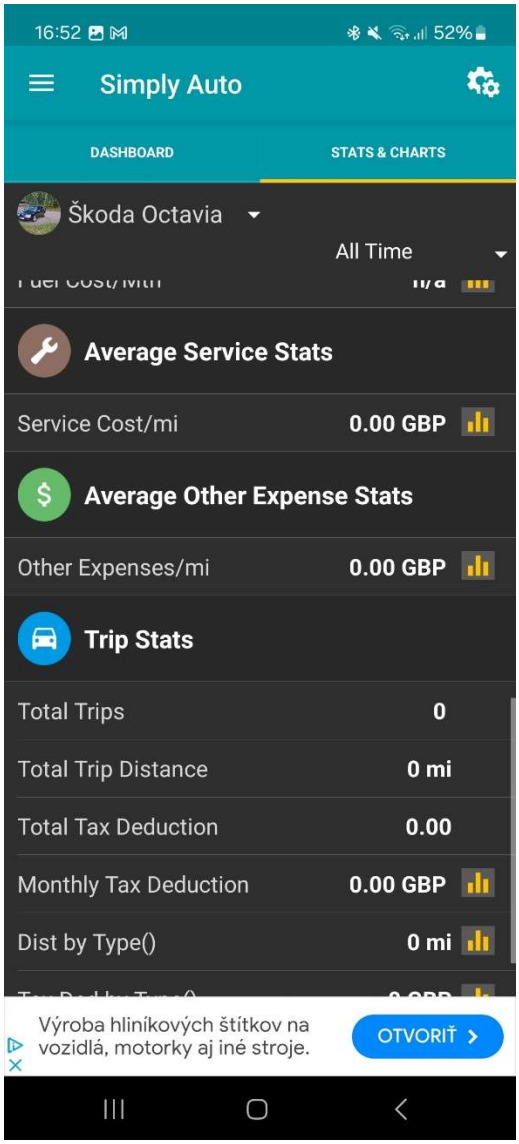


Servisné pripomienky



Celková štatistika





Aplikácia CAR-MAIN sa líši tým, že ponúka zobrazenie výdavkov na časovej osi, čo umožňuje prehľadné zobrazenie v časovom horizonte

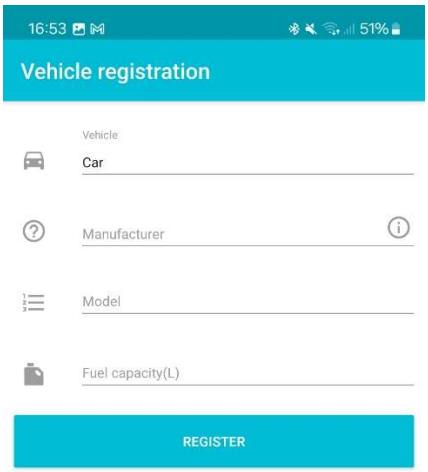
DRIVVO

Táto aplikácia je veľmi pekne graficky spracovaná a ponúka rozšírené funkcionality za príplatok za premium verziu.

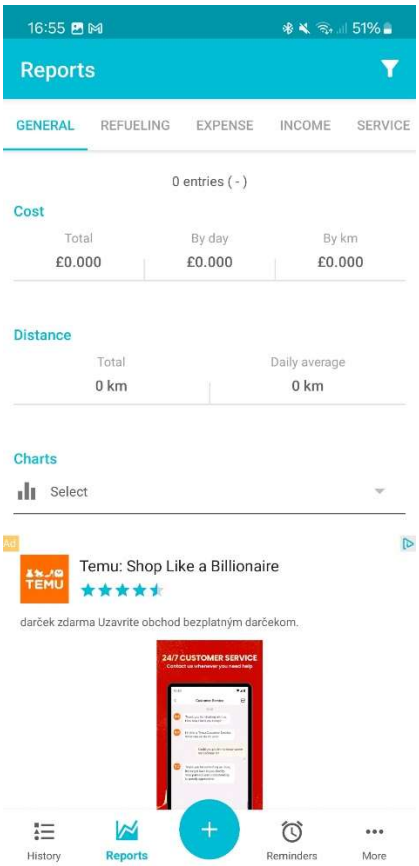
Výhody:

- Grafické spracovanie
- Časová línia

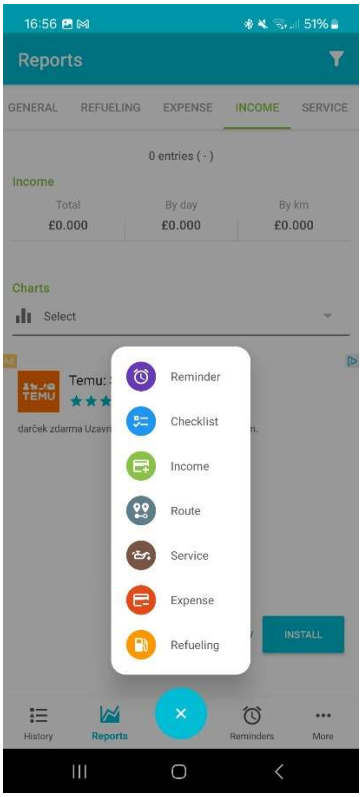
Nevýhody: Prehľad cien palív čerpacích staníc je neaktuálny, ale nápad dobý

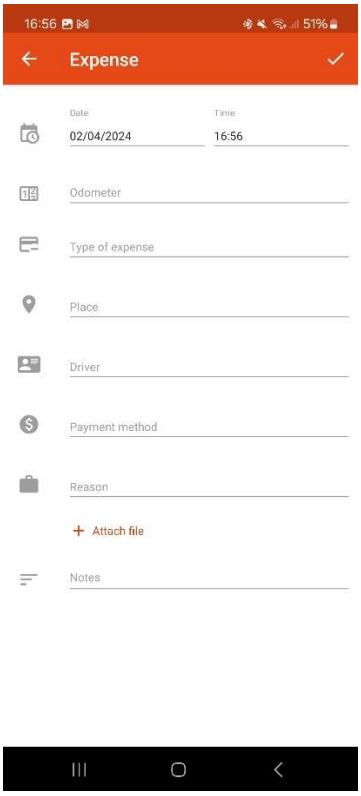


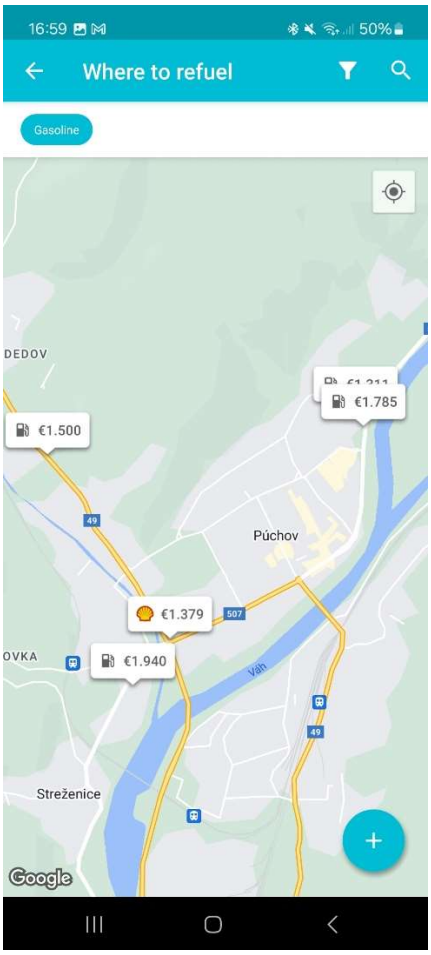
Vkladanie údajov vozidla

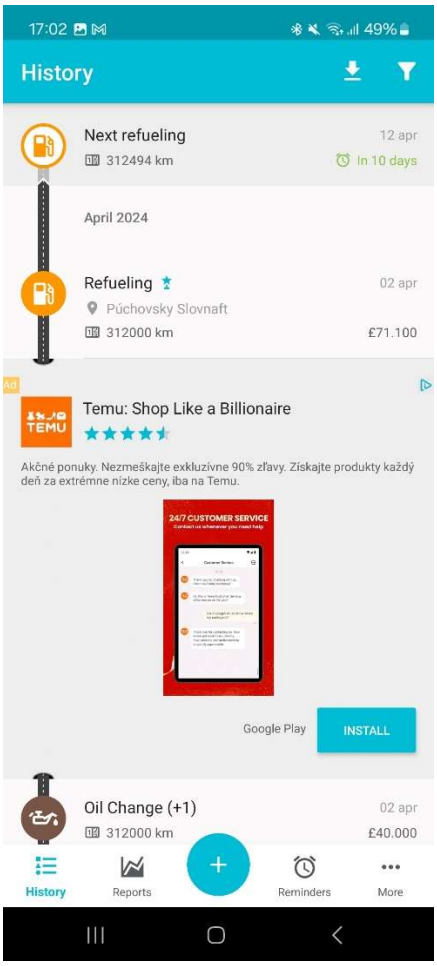


Prehľad údajov









Aplikácia Drivvo vie byť trochu komplikovaná pre niektorých používateľov a jednoduchšie ovládanie aplikácie CAR-MAIN by mohlo byť vhodnejšie pre niektorých používateľov

CARDIARY

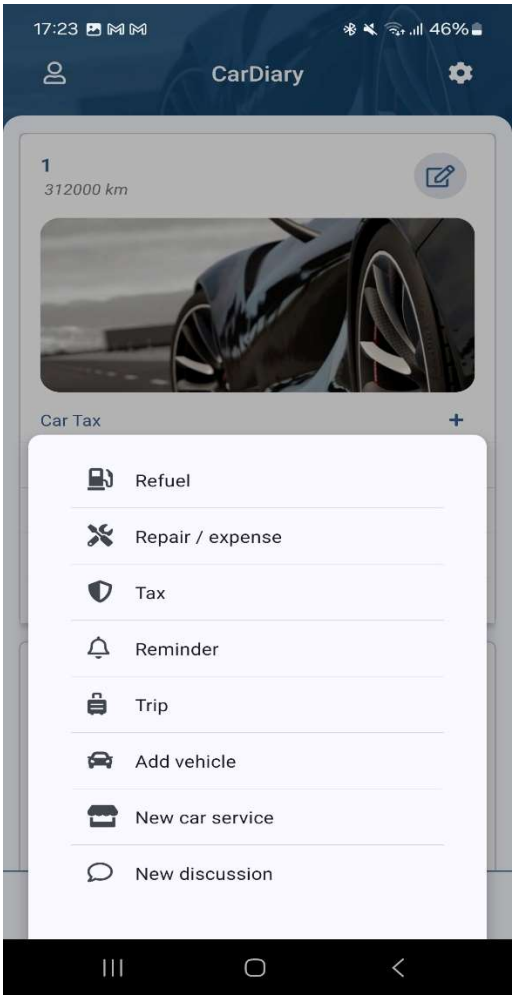
Poslednou aplikáciou je CarDiary, ktorá ponúka opäť pekné grafické vyhotovenie a intuitívne ovládanie. Ponúka taktiež aj platené verzie s rozšírenými možnosťami.

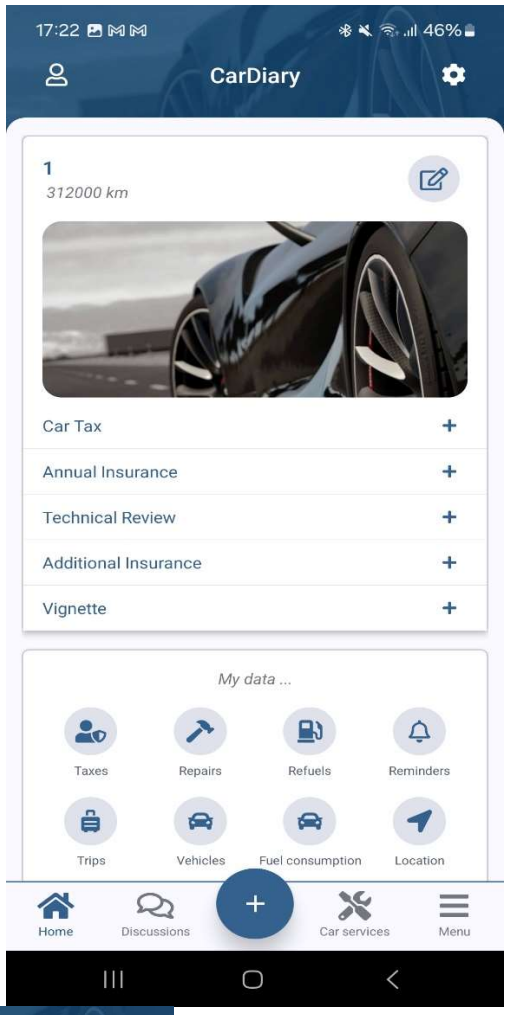
Výhody:

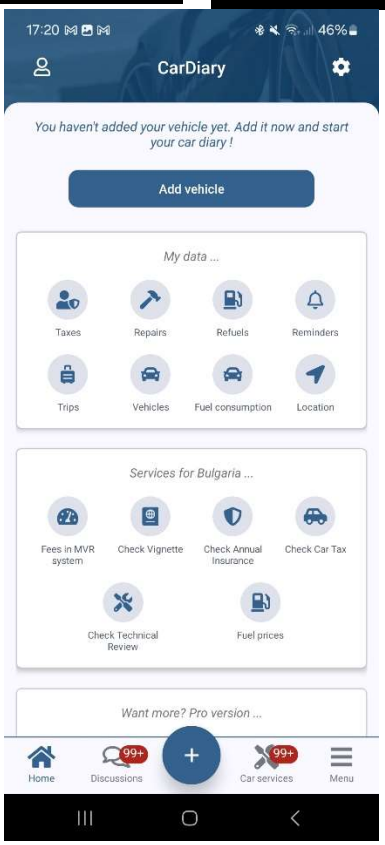
- Diskusia s komunitou používateľov
- Ponuky automobilových servisov

Nevýhody:

- Diskusia a možnosti servisu sú v podstate nepoužiteľné, pretože sú v rôznych krajinách a jazykoch.


17:23
46%
CarDiary
1
312000 km
Car Tax
Refuel
Repair / expense
Tax
Reminder
Trip
Add vehicle
New car service
New discussion


17:22
46%
CarDiary
1
312000 km
Car Tax
Annual Insurance
Technical Review
Additional Insurance
Vignette
My data ...
Taxes
Repairs
Refuels
Reminders
Trips
Vehicles
Fuel consumption
Location
Home
Discussions
Car services
Menu

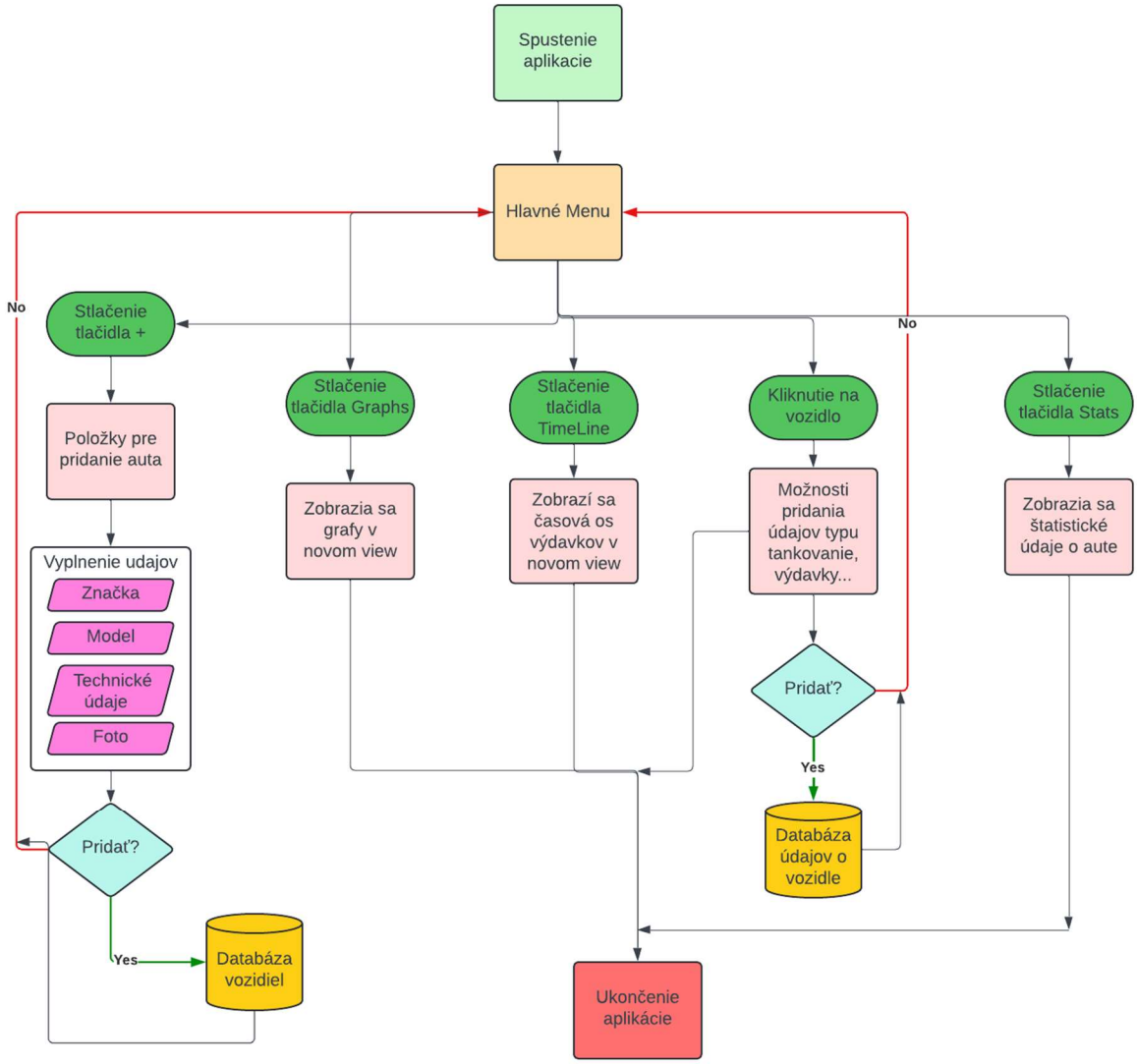

17:20
46%
CarDiary
You haven't added your vehicle yet. Add it now and start your car diary !
Add vehicle
My data ...
Taxes
Repairs
Refuels
Reminders
Trips
Vehicles
Fuel consumption
Location
Services for Bulgaria ...
Fees in MVR system
Check Vignette
Check Annual Insurance
Check Car Tax
Check Technical Review
Fuel prices
Want more? Pro version ...
Home
Discussions
Car services
Menu

Aplikácia CarDiary neponúka zobrazenie na časovej osi a viacero funkcionalít je za príplatok. Niektoré funkcie sú aj zbytočné a používateľ ich nemusí využiť, preto by použitie aplikície Car-Main mohlo byť vhodnou alternatívou.

2) Návrh riešenia problému

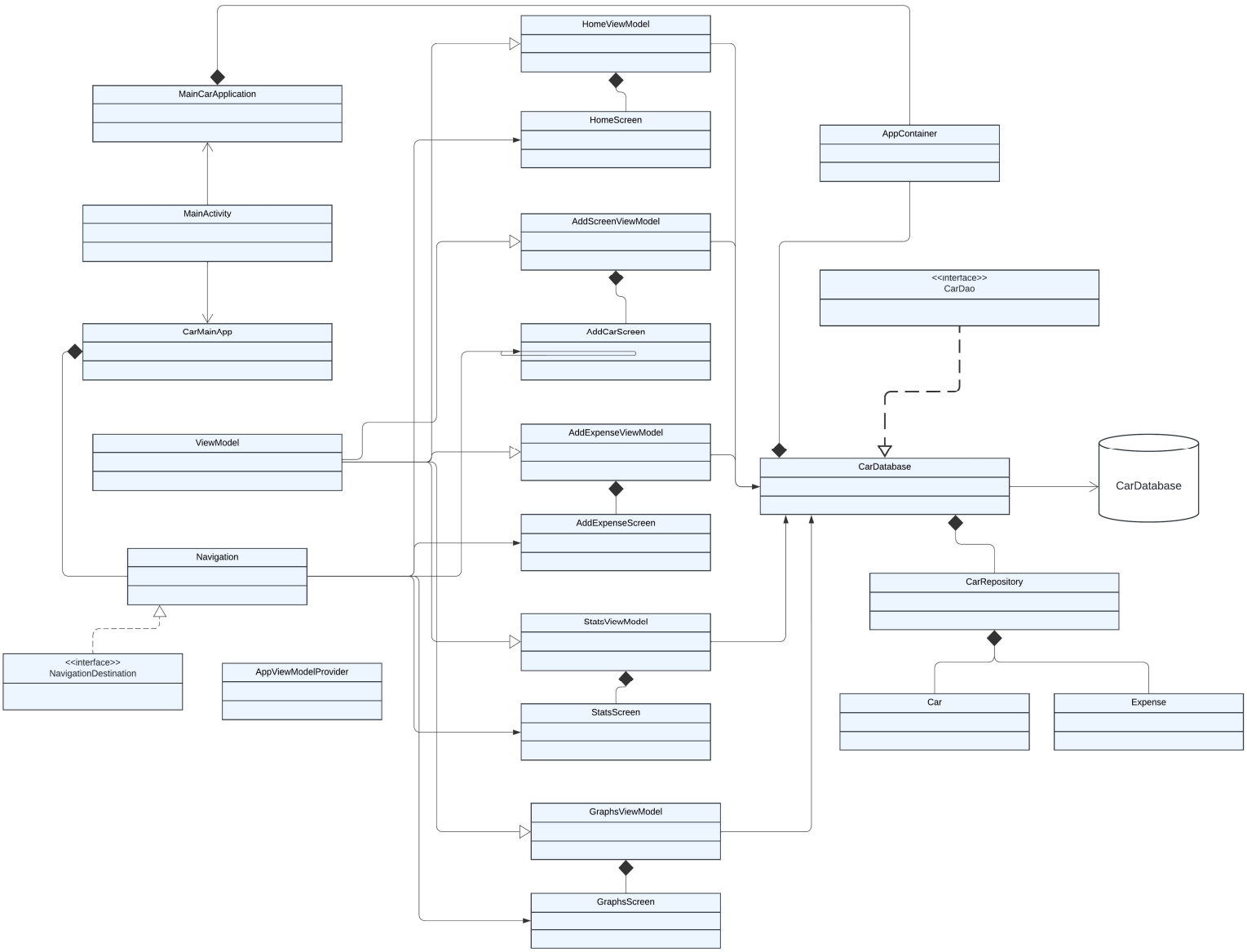
Pre daný projekt sme si vytvorili UML diagram tried a prehľad procesov, ktoré by aplikácia mala spĺňať a ako by mali procesy prebiehať.

2.1) Prehľad procesov aplikácie



Obrázok 1 prehľad procesov aplikácie

2.2) UML diagram tried



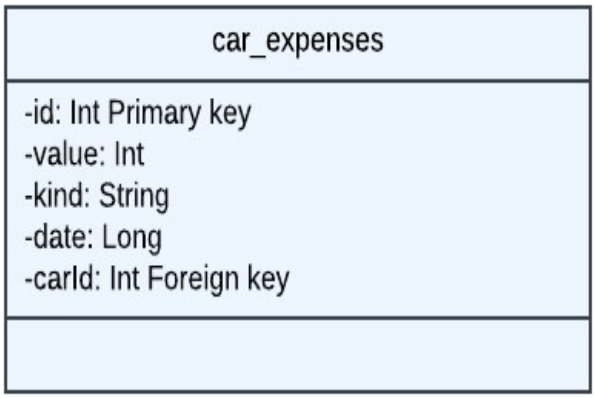
Obrázok 2 UML diagram tried

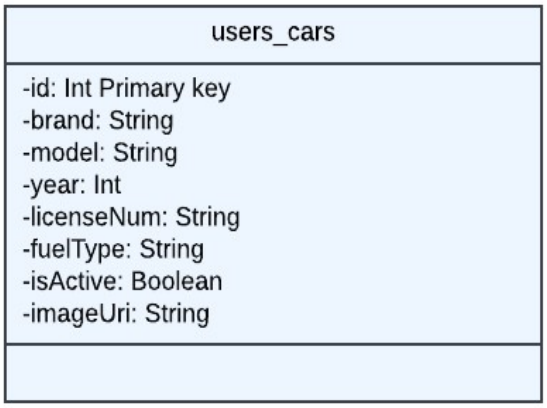
3) Popis implementácie

Na implementáciu projektu som zvolil MVVW architektúru. Každá obrazovka (View) má svoj vlastný ViewModel pre komunikáciu s databázou.

3.1) Použitie databázy- Room

V databáze sa nachádzajú dve tabuľky: users_cars, car_expenses





Príkazy pre komunikáciu s databázou sú v rozhraní CarDao. Sú tu operácie pre vkladanie, dotazovanie, upravovanie a mazanie údajov z tabuliek.

Trieda CarDatabase je návrhový vzor singleton a vracia inštanciu databázy.

Trieda CarsRepository obsahuje metódy pre komunikáciu s databázou (sú tu všetky metódy, ktoré sú aj v rozhraní CarDao) a **trieda OfflineCarRepositoy** dedí od tejto triedy a sprostredkováva interakciu medzi databázou a aplikáciou.

3.2) Použitie navigácie- Navigation

Navigácia v programe prebieha prostredníctvom ciest (routes), kde každá obrazovka má vlastnú unikátnu cestu a zoznam parametrov, ktoré potrebuje (ak nejaké potrebuje).

Trieda Navigation obsahuje object NavHost, ktorý spravuje prechod medzi obrazovkami a predávanie parametrov. Ukážka navigácie a posielania parametrov:

```
**
 * navigation for TimeLine screen
 */
composable(
    route = TimeLineDestination.routeWithArgs,
    arguments = listOf(navArgument(TimeLineDestination.carIdArg) {
        type = NavType.IntType
    })
){
        TimeLineScreen(
            navigateBack = { navController.navigate(HomeDestination.route)
},
            navController = navController)
}
```

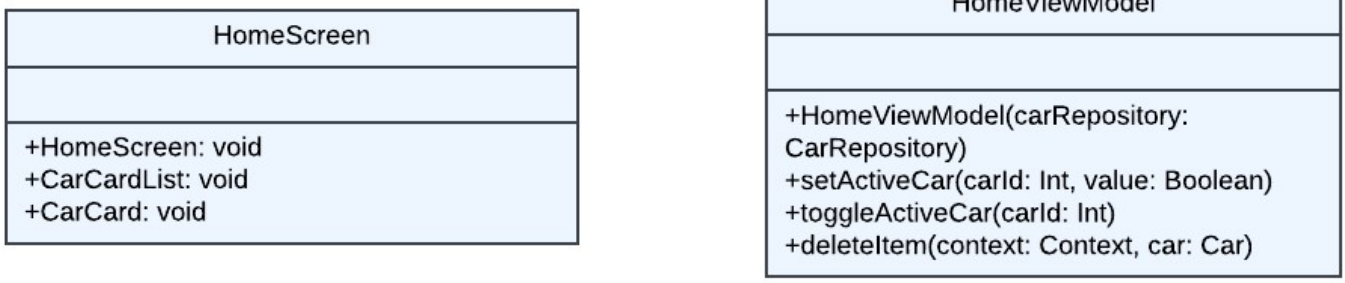
3.3) Použitie ViewModel

Ako už bolo vyššie spomenuté, každá obrazovka má svoj vlastný ViewModel. O inicializáciu každého ViewModelu sa stará **trieda AppViewModelProvider**. Táto trieda je typu singleton a obsahuje inštancie všetkých ViewModelov. Bližší popis bude uvedený pre konkrétne obrazovky.

3.4) HomeScreen a HomeViewModel

HomeScreen obsahuje interaktívne kartové zobrazenie všetkých používateľových áut. Umožňuje pridávanie ďalších vozidiel pomocou floating buttonu “Add”. Odoberanie prebieha kliknutím na tlačidlo “trash”. To vozidlo, ktoré ma na karte zobrazený nápis Active je posielané ako parameter do ďalších obrazoviek.

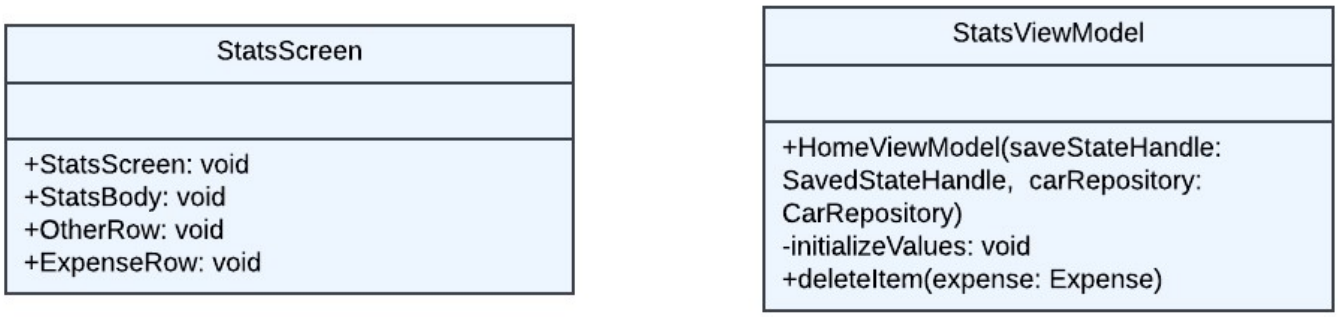
Trieda HomeScreenViewModel obsahuje metódy pre komunikáciu s databázou. Prebieha tu pridávanie vozidiel, zmena údajov a odstraňovanie údajov.



3.5) StatsScreen a StatsViewModel

StatsScreen obsahuje tabuľkové zobrazenie všetkých výdavkov o vozidle a štatistické údaje o celkovej sume pre jednotlivé druhy.

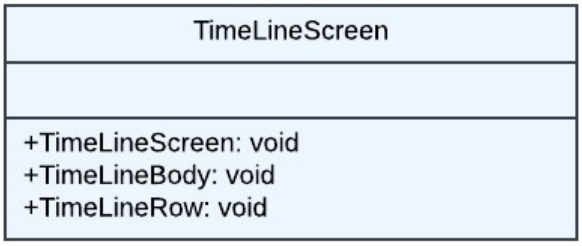
Trieda StatsViewModel obsahuje údaje o výdavkoch, ktoré preberá z databázy. Umožňuje tiež odstraňovať výdavky.

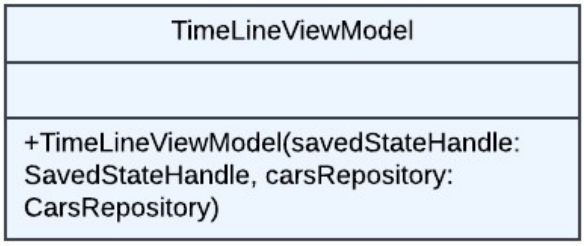


3.6) TimeLineScreen a TimeLineViewModel

TimeLineScreen obsahuje časovú os, na ktorej sú zachytené všetky výdavky. Implementácia je pomocou Row.

Trieda TimeLineViewModel sprostredkováva údaje pre zobrazenie na časovej osi.



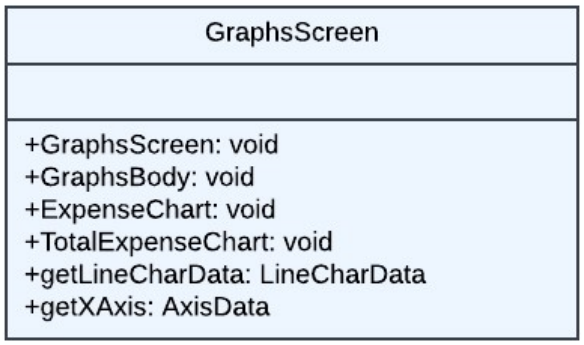


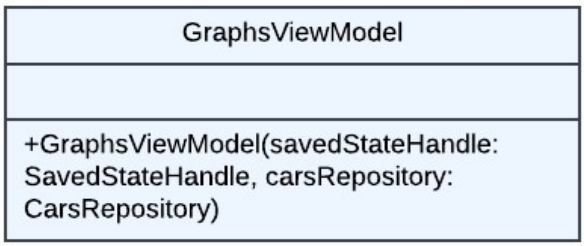
3.7) GraphsScreen a GraphsViewModel

GraphsScreen obsahuje grafické zobrazenie celkových výdavkov, ale aj jednotlivo.

Trieda GraphsViewModel zabezpečuje údaje o výdavkoch pre grafy.

Na vyobrazenie grafov je použitá externá knižnica YCharts, kde je použitý LineGraph (https://github.com/codeandtheory/YCharts)

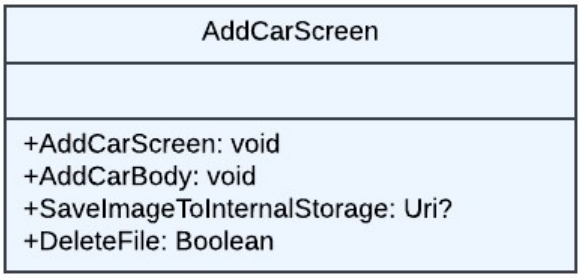


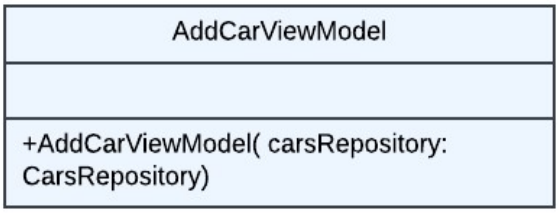


3.8) AddCarScreen a AddCarViewModel

AddCarScreen obsahuje textové polia kde používateľ zadáva potrebné údaje o vozidle a nahráva fotografiu.

Trieda AddCarViewModel obsahuje metódy pre vkladanie záznamov do tabuliek, ale predtým skontroluje či sú zadané údaje validné.

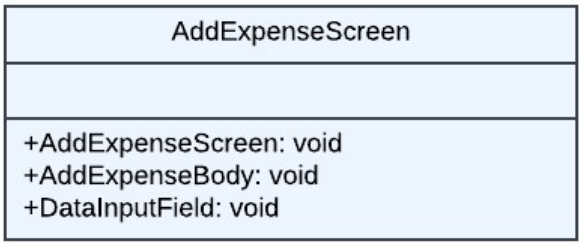


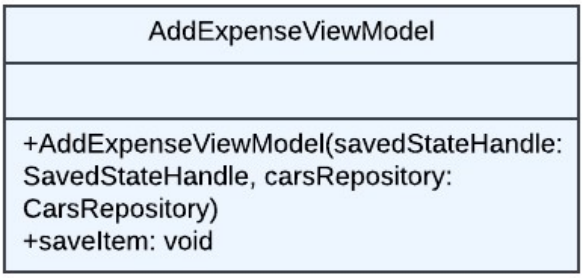


3.9) AddExpenseScreen a AddExpenseViewModel

AddExpenseScreen obsahuje polia pre zadanie potrebných údajov pre vloženie nového výdavku.

Trieda AddExpenseViewModel zabezpečuje vloženie nového výdavku do databázy.

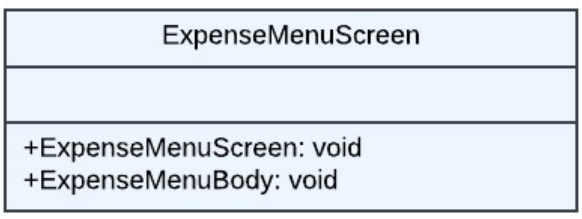


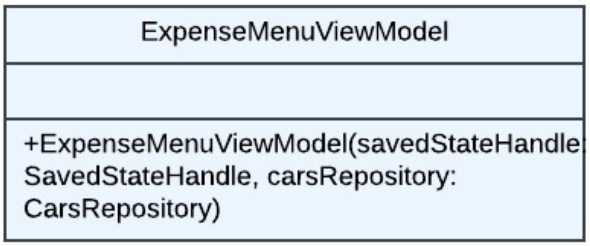


3.10) ExpenseMenuScreen a ExpenseMenuViewModel

ExpenseMenuScreen ponúka na výber pridanie viacerých druhov výdavkov podľa stlačenia tlačidla.

Trieda ExpenseMenuViewModel zabezpečuje komunikáciu s databázou.





4) Zoznam použitých zdrojov

Práca s databázou:

https://developer.android.com/codelabs/basic-android-kotlin-compose-persisting-data-room#0

https://youtu.be/hrJZIF7qSSw?si=FIXJKindLUQWYKXb - migrácia medzi verziami

Viacero komponentov som použil z nasledujúceho codelabu (hlavne databázu, AppViewModelProvider, pridávanie itemov a Scafold layout):

https://developer.android.com/codelabs/basic-android-kotlin-compose-persisting-data-room#0

Veľa kódu som upravoval alebo vytváral pomocou AI (chatgpt)

Grafy použité v GraphsScreen:

https://github.com/codeandtheory/YCharts