TankApp

Technikai dokumentáció

Fejlesztők:
Dóczi Dávid
Egyed Endre
Habony Zoltán
Kovács Adrián

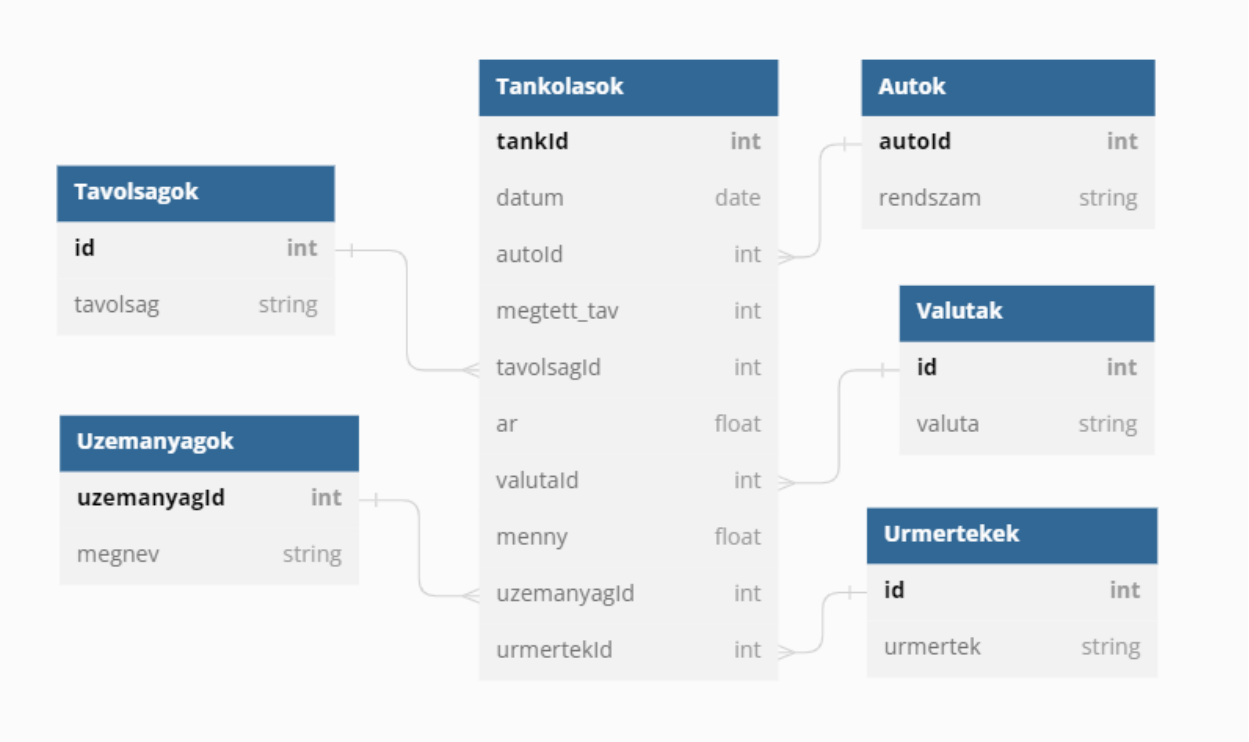
Tartalom

Rendszerkövetelmények

Operációs rendszer	Android 8.0 és újabb
Szabad tárhely	18,4 MB
Alkalmazás engedélyek	nincs

Adatbázis

Adatbázisterv



Változás a tervhez képest: Tankolasok.datum mező long típusú (1970.01.01 óta eltelt napok számát tárolja), és az Autok táblában a „megj” mező.

Tavolsagok tábla

Mező neve	Típusa	Szerepe, jelentése
id	INTEGER, AUTO INCREMENT	Primary Key
tavolsag	TEXT	Távolság mértékegység pl „km”

Kódbeli modellje: data.models. TavolsagModel

Uzemanyagok tábla

Mező neve	Típusa	Szerepe, jelentése
uzemanyagId	INTEGER, AUTO INCREMENT	Primary Key
megnev	TEXT	Üzemanyag típus pl „biodízel”

Kódbeli modellje: data.models. UzemanyagModel

Autok tábla

Mező neve	Típusa	Szerepe, jelentése
autoId	INTEGER, AUTO INCREMENT	Primary Key
rendszam	TEXT	Jármű rendszáma
megj	TEXT	Opcionális megjegyzés a járműhöz a könnyebb beazonosítás érdekében

Kódbeli modellje: data.models. AutoModel

Valutak tábla

Mező neve	Típusa	Szerepe, jelentése
id	INTEGER, AUTO INCREMENT	Primary Key
valuta	TEXT	Valuta rövid neve pl „HUF”

Kódbeli modellje: data.models. ValutaModel

Urmertekek tábla

Mező neve	Típusa	Szerepe, jelentése
id	INTEGER, AUTO INCREMENT	Primary Key
urmertek	TEXT	Űrmérték mértékegység

Kódbeli modellje: data.models. UrmertekModel

Tankolasok tábla

Mező neve	Típusa	Szerepe, jelentése
tankId	INTEGER, AUTO INCREMENT	Primary Key
datum	INTEGER	Tankolás időpontja, 1970.01.01-től eltelt napok száma
autoId	INTEGER	Foreign Key az Autok táblához
megtett_tav	INTEGER	Előző tankolás óta megtett távolság kilométerben vagy mérföldben
tavolsagId	INTEGER	Foreign Key a Tavolsagok táblához
ar	FLOAT	Tankolt üzemanyag egységára
valutaId	INTEGER	Foreign Key a Valutak táblához
menny	FLOAT	Tankolt üzemanyag mennyisége
uzemanyagId	INTEGER	Foreign Key az Uzemanyagok táblához
urmertekId	INTEGER	Foreign Key az Urmertekek táblához

Kódbeli modellje: data.models. TankolasModel
Megjegyzés: A tábla adatait data. TankolasOsszetett objektumokkal dolgozzuk fel (ez minden tábla összekapcsolásának a leképezése)

Navigációs felület

mobile_navigation.xml

A **mobile_navigation.xml** leírja az alkalmazás fragment-jeinek kapcsolatait és az azok közötti navigációt. Az xml állomány tartalmaz egy **<navigation>** tag-et, melyben meghatározzuk az Android névterét, a MainActivity-t **<activity>**, az alkalmazás indításának kezdetekor megjelenő fragment-et, az alkalmazás életciklusában előforduló fragment-eket **<fragment>** és azok kapcsolatait **<action>**.

Kódrészlet

```
<?xml version="1.0" encoding="utf-8"?>
<navigation xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/mobile_navigation"
    app:startDestination="@+id/nav_kezdo">

    <fragment
        android:id="@+id/nav_kezdo"
        android:name="com.example.tankapp.ui.kezdo.KezdoFragment"
        android:label="Kezdőlap"
        tools:layout="@layout/fragment_kezdo" >
        <action
            android:id="@+id/action_nav_kezdo_to_nav_jarmuvek"
            app:destination="@id/nav_jarmuvek" />
```

```
    </fragment>

    <activity

        android:id="@+id/mainActivity"

        android:name="com.example.tankapp.MainActivity"

        android:label="fragment_tankolas_felvetel"

        tools:layout="@layout/fragment_tankolas_felvetel" />

</navigation>
```

activity_main_drawer.xml

Az **activity_main_drawer.xml** felel a mobil menü megjelenésének leírásáért. A **<menu>** tag tartalmazza a menü elemeket **<item>** egy csoportba foglalva **<group>**. Az adott menüponthoz hozzá van rendelve alapértelmezetten egy ikon **android:icon** és egy menüpont elnevezés is **android:title**.

Kódrészlet

```
<menu xmlns:android="http://schemas.android.com/apk/res/android"

    xmlns:tools="http://schemas.android.com/tools"

    tools:showIn="navigation_view">



    <group android:checkableBehavior="single">

        <item

            android:id="@+id/nav_kezdo"

            android:icon="@drawable/baseline_home_24"

            android:iconTint="@android:color/white"

            android:title="@string/menu_home_page" />
    </group>

</menu>
```

activity_main.xml

Az **activity_main.xml** írja le az alkalmazás megjelenésének fő részeit.

- Magába foglalja az alkalmazás tetején megjelenő sávot **app_bar_main.xml**, ami tartalmazza az alkalmazás nevét és a menü előhozásáért felelős menü ikont. Továbbá az **app_bar_main.xml** tartalmaz egy területet is, amibe a különböző funkciók megvalósításáért felelős oldalak kerülnek majd betöltésre.
- Tartalmaz egy navigációs nézetet, melyben leírásra kerül a **nav_header_main.xml** és az **activity_main_drawer.xml** alapján a menü.

Kódrészlet

```
<?xml version="1.0" encoding="utf-8"?>

<androidx.drawerlayout.widget.DrawerLayout
xmlns:android="http://schemas.android.com/apk/res/android"

    xmlns:app="http://schemas.android.com/apk/res-auto"

    xmlns:tools="http://schemas.android.com/tools"

    android:id="@+id/drawer_layout"

    android:layout_width="match_parent"

    android:layout_height="match_parent"

    android:fitsSystemWindows="true"

    tools:openDrawer="start">



    <include

        android:id="@+id/app_bar_main"

        layout="@layout/app_bar_main"

        android:layout_width="match_parent"

        android:layout_height="match_parent" />



    <com.google.android.material.navigation.NavigationView

        android:id="@+id/nav_view"

        android:layout_width="wrap_content"

        android:layout_height="match_parent"

        android:layout_gravity="start"

        android:background="@color/black"

        android:fitsSystemWindows="true"

        app:headerLayout="@layout/nav_header_main"

        app:itemIconTint="@android:color/white"

        app:itemTextColor="@android:color/white"
```

```
        app:menu="@menu/activity_main_drawer" />

</androidx.drawerlayout.widget.DrawerLayout>
```

Tankolás felvétel

fragment_tankolas_felvetel.xml

A fragment_tankolas_felvetel.xml írja le azt az űrlapot, melyen a felhasználó képes új tankolásokat rögzíteni.

Vezérlőelemek

Vezérlőelem típusa	Vezérlőelem azonosítója	Tartalmazott adat típusa / funkció
TextView	mainTitle	String / szöveges megjelenítés
TextView	dateTitle	String / szöveges megjelenítés
TextView	dateAlertText	String / hibaüzenet megjelenítés
Button	setDateTimeButton	String / Dátum megjelenítés, kiválasztás
TextView	distanceTitle	String / szöveges megjelenítés
TextView	distanceAlertText	String / hibaüzenet megjelenítés
EditText	distanceInputField	Number / érték megadás
Spinner	distanceUnitSpinner	TavolsagModel.tavolsag / mértékegység kiválasztás
TextView	fuelQuantityTitle	String / szöveges megjelenítés
TextView	fuelUnitTitle	String / szöveges megjelenítés
TextView	fuelQuantityAlertText	String / hibaüzenet megjelenítés
Spinner	fuelTypeSpinner	UzemanyagModel.Megnev / üzemanyag típus megnevezés
EditText	fuelQuantityInputField	Number / üzemanyag egység megadás
Spinner	fuelUnitTypeSpinner	UrmertekModel.Urmertek / üzemanyag mértékegység megnevezés
TextView	fuelPriceTitle	String / szöveges megjelenítés
TextView	fuelPriceAlertText	String / hibaüzenet megjelenítés
EditText	fuelPriceInputField	Number / üzemanyag egységár megadás
TextView	fuelCurrencyTitle	String / szöveges megjelenítés
Spinner	fuelCurrencySpinner	ValutaModel.Valuta / valuta megadása
Button	saveRefuellingButton	tankolás mentése

TankolasFelvetelFragment.java

A TankolasFelvetelFramgent.java osztály felel a feleületen történő események és funkciók megvalósításáért. Itt definiáljuk, hogy az adott vezérlőelemek, hogyan és miként jöjjenek létre az alkalmazás indulásakor. Itt kapcsoljuk össze a felületen szereplő vezérlő elemeket a programban szereplő változókkal.

setDateTimeButton (btn_date)

A **setDateTimeButton OnClick** metódusa, akkor hívódik, amikor egy kattintás esemény be nem következik. A bekövetkezéskor létrejön egy **Calendar** osztály típusú példány, amit a dátumkiválasztó párbeszéd ablak konstruktorának adunk át paraméterül. Ezzel biztosítva, hogy a naptárban az éppen aktuális napnak megfelelő időpont legyen kiválasztva. **DatePickerDialog.OnDateSetListener()** metódus segítségével a párbeszédablakban történő kiválasztás után, mi történjen. Tartalmaz egy kötelezően kifejtendő metódust az **onDataSet** metódust. Itt meg lett adva, hogy a kiválasztás után a dátum értékét állítsa be a **setDateTimeButton Text** tulajdonság értékének.

Járművek, jármű felvétel

jarmu_lista_elem.xml

A **jarmu_lista_elem.xml** felel a listában szereplő járművek általános megjelenítésének leírásáért. Minden egyes jármű adatai egy ilyen komponensben vannak ábrázolva. Ezeket a komponenseket egy **CardView** nézet foglalja magába. A **CardView** tartalmaz minden más

vezérlőelemet, amit meg kell jeleníteni.

Kódrészlet

```
<?xml version="1.0" encoding="utf-8"?>

<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"

    xmlns:app="http://schemas.android.com/apk/res-auto"

    xmlns:tools="http://schemas.android.com/tools"

    android:layout_width="match_parent"

    android:layout_height="wrap_content"

    android:paddingBottom="15dp">


   <androidx.cardview.widget.CardView

        android:layout_width="match_parent"

        android:layout_height="wrap_content"


        app:layout_constraintStart_toStartOf="parent"

        app:layout_constraintTop_toTopOf="parent">


    <LinearLayout

        android:id="@+id/linearLayout"

        android:padding="@dimen/activity_horizontal_margin"

        android:orientation="vertical"

        android:background="@color/bottomBar"

        android:layout_width="match_parent"

        android:layout_height="wrap_content">


         <TextView

             android:id="@+id/numberPlateTitle"

             android:text="Heading"

             android:textColor="@color/white2"

             android:textAllCaps="true"

             android:textStyle="bold"

android:textAppearance="@style/TextAppearance.Material3.TitleLarge"

             android:layout_width="wrap_content"
```

```
            android:layout_height="wrap_content">

        </TextView>
    </LinearLayout>

   </androidx.cardview.widget.CardView>
</androidx.constraintlayout.widget.ConstraintLayout>
```

fragment_jarmuvek.xml

A **fragment_jarmuvek.xml** tartalmazza a listás nézetet az adatbázisban szereplő járművek adatairól. A felületen lehetőség van egy új oldalra navigálni és újabb járműveket felvenni. Ezt egy gomb segítségével érhetjük el **addCarButton**. A listás megjelenítésért egy **RecyclerView** felelős, ami a **jarmu_lista_elem.xml** felhasználásával legenerálja az adott elemeket a felületen megjelenő vezérlőbe.

JarmuvekAdapter

A **JarmuvekAdapter** osztály és azok metódusai segítségével generálódik le az adat, jön létre a megfelelő nézet és a hozzá tartozó információ.

- A **JarmuvekAdapter** osztály bővítettük a **RecyclerView.Adapter<JarmuAdapter.Viewholder>**-el. Ez az adapter kapcsolatot biztosít egy alkalmazásspecifikus adatkészletből az alkalmazáson belül megjelenített nézetekhez.
- Létre lett hozva egy **ViewHolder** osztály és ki lett terjesztve **RecyclerView.ViewHolder** absztrakt osztállyal, ami implementálja a **ViewHolder** metódust. Ebben a metódusban kapcsoljuk össze a felületen megjelenő vezérlőelemeket, az osztályban szereplő adattagokkal.
- A **RecyclerView.Adapter<JarmuAdapter.Viewholder>** van egy **OnCreateViewHolder** kötelezően kidolgozandó metódusa. Ebben a metódusban jön létre az a **ViewHolder**, ami tartalmazni fogja a járművek adatait, a **jarmu_lista_elem.xml** kinézet alapján.
- A **RecyclerView.Adapter<JarmuAdapter.Viewholder>** van egy **onBindViewHolder** metódusa. Ebben a metódusban történik meg az adatkötés a felület és a jármű adatai között.
- A **getItemCount** függvény visszaadja a RecyclerView hoz használt lista méretét.

JarmuvekFragment.java

A JarmuvekFragment.java osztályban valósul meg a vezérlő elemek hozzárendelése a változókhoz és a az adapter beállítása a **RecyclerView** adapter tulajdonságának. Emellett a felületen meghatározott **addCarButton onClick** eseménye is definiálva lett, amely átirányít az **JarmuFelvetelFragment** felületre.

fragment_jarmu_felvetel.xml

A **fragment_jarmu_felvetel.xml** tartalmazza a jármű felvétel űrlapjának nézetét. A járművek oldalról lehet, az új jármű hozzáadása gombbal a nézetre navigálni.

Vezérlőelemek

Vezérlőelem típusa	Vezérlőelem azonosítója	Tartalmazott adat típusa / funkció
TextView	mainTitle	String / szöveges megjelenítés
TextView	numberPlateTitle	String / szöveges megjelenítés
TextView	numberPlateAlertText	String / hibaüzenet megjelenítés
EditText	numberPlateInputField	Text / Rendszám megadása
TextView	carNameTitle	String / szöveges megjelenítés
TextView	carNameAlertText	String / hibaüzenet megjelenítés
EditText	carNameInputField	Text / Megjegyzés megadás
Button	saveVehicleButton	jármű mentése

Kezdőoldal

fragment_kezdo.xml

A **fragment_kezdo.xml** tartalmazza a legutolsó tankolást, az átlagfogyasztást és az összes megtett út adatait valamint a statisztikákat a heti és havi bontásokról.

Vezérlőelem típusa	**Vezérlőelem azonosítója**	**Tartalmazott adat típusa / funkció**
TextView	osszesut_text	String / szöveges megjelenítés
TextView	atlagfogyasztas	String / szöveges megjelenítés
TextView	km_utan	String / szöveges megjelenítés
TextView	uzemanyag_tipus	String / szöveges megjelenítés
TextView	uzemanyag_ar	String / szöveges megjelenítés
TextView	uzemanyag_mennyiseg	String / szöveges megjelenítés
TextView	textView19	String / szöveges megjelenítés
TextView	eltelt_ido	String / szöveges megjelenítés
TextView	atlag_fogyasztas_text	String / szöveges megjelenítés
TextView	osszesut	String / szöveges megjelenítés
TextView	statisztikaCim	String / szöveges megjelenítés
TextView	statisztikaHeti	String / szöveges megjelenítés
TextView	statisztikaHavi	String / szöveges megjelenítés
BarChart	barChart	Statisztika megjelenítése

KezdoFragment.java

A KezdoFragmant.java felel a kezdőlap adatainak megjelenítéséért, ez magába foglalja a vezérlő elemek és a diagrammok elkészítését is.

- A **weeklyChart** és a **monthlyChart** szövegnek van egy-egy **OnClick** eseménye, amely megváltoztatja a grafikon tartalmát az adott bontásnak megfelelően. Ebben szerepel **createBarData** metódus, ami a bontásoknak megfelelő adatok segítségével elkészíti az oszlopdiagrammokat.
- A **getDataFromStatistics** metódus Összegyűjti az X és az Y érték adatait az oszlopdiagramhoz. paramétere egy **TankolasokSzamaBontasban** típusú lista, ami tartalmazza az x és y értékeket. Visszatérési értéke egy lista, ami visszaadja az adott oszlophoz szükséges adatokat.
- A **createBarData** metódus előállítja a diagram elkészítéséhez szükséges oszlopokat
- Paramétere egy lista ami tartalmazza az oszlopok értékeit, egy szöveg, ami a grafikon címkéjének neve pl:heti bontás. Visszatérési értéke egy **BarData** adathalmaz, ami majd a grafikon ábrázolásához szükséges.

Adatbázis kezelés

DatabaseHelper.java

Ez az osztály eredetileg arra a célra lett létrehozva, hogy kezelje az adatbázist. Ha nincs létrehozva adatbázis, akkor azt létrehozza. Ebben a megvalósításban szükséges volt ezt az osztályt örököltetni azSQLiteOpenHelper osztálytól, és implementálni az onCreate és az onUpgrade metódusokat. Ezek közül az onUpgrade-et nem igazán használtuk. Ez a metódus akkor lett volna releváns, ha új (nagyobb)verziószámokat adunk meg a konstruktorban a szülő konstruktorának paraméterként, tehát növeljük az adatbázis verziószámát. Az onCreate metódus viszont létrehozza az adatbázist, ha az még nincs
létrehozva. Az ehhez szükséges SQL parancsok az execSQL függvény segítségével lettek betáplálva.

```
public void onCreate(SQLiteDatabase db) {
    db.execSQL("CREATE TABLE Autok (autoId INTEGER PRIMARY KEY AUTOINCREMENT, rendszam TEXT UNIQUE, megj TEXT)"); //2db
    db.execSQL("CREATE TABLE Valutak (id INTEGER PRIMARY KEY AUTOINCREMENT, valuta TEXT)"); //huf, Usd
    db.execSQL("CREATE TABLE Urmertekek (id INTEGER PRIMARY KEY AUTOINCREMENT, urmertek TEXT)"); //l,gl
    db.execSQL("CREATE TABLE Tavolsagok (id INTEGER PRIMARY KEY AUTOINCREMENT, tavolsag TEXT)"); //km, miles
    db.execSQL("CREATE TABLE Uzemanyagok (uzemanyagId INTEGER PRIMARY KEY AUTOINCREMENT, megnev TEXT)"); //benzin, diesel
    db.execSQL("CREATE TABLE Tankolasok (tankId INTEGER PRIMARY KEY AUTOINCREMENT, datum INTEGER, autoId INTEGER, megtett_tav I
```

```
DatabaseHelper
```

```
# db : SQLiteDatabase
+ DEFAULT_DBNAME : String {readOnly}
```

```
+ DatabaseHelper()
+ DatabaseHelper(dbName : String)
+ onCreate(db : SQLiteDatabase) : void
- feltolt() : void
- init() : void
+ dbTest() : void
+ onUpgrade(db : SQLiteDatabase, i : int, i1 : int) : void
+ addAutok(Rendszam : String, megj : String) : void
+ addValutak(Valuta : String) : void
+ addUrmertekek(Urmertek : String) : void
+ addTavolsagok(Tavolsag : String) : void
+ addUzemanyagok(Megnev : String) : void
+ addTankolasok(Datum : long, AutoId : int, Megtett_tav : int, TavolsagId : int, Ar : float, ValutaId : int, Menny : float, UzemanyagId : int
+ getOsszesTankolas() : ArrayList<TankolasOsszetett>
+ getAutok() : ArrayList<AutoModel>
+ getUtolsoTankolas() : TankolasOsszetett
+ getTankolasokSzama() : int
+ getUzemanyagok() : ArrayList<UzemanyagModel>
+ getValutak() : ArrayList<ValutaModel>
+ getUrmertekek() : ArrayList<UrmertekModel>
+ getTavolsagok() : ArrayList<TavolsagModel>
+ getTankolasokByAutoId(autoId : int) : ArrayList<TankolasOsszetett>
+ getDatumokByAutoId(autoId : int) : ArrayList<LocalDate>
+ getJarmuvekSzama() : int
+ kiurit() : void
+ getDatabaseName() : String
+ getDbDirectory() : String
```

TA-3: Tesztadatokkal feltöltés

DatabaseHelper.java:

- A tesztadatok megadásához létrehoztam minden, a tankolás megadásával kapcsolatosadathoz egy metódust, melyek beszúrnak adatokat az adatbázisba

```java
public void addAutok(String Rendszam, String megj) {
    ContentValues values = new ContentValues();
    values.put("rendszam", Rendszam);
    values.put("megj", megj);
    db.insertOrThrow( table: "Autok",  nullColumnHack: null, values);
}

2 usages  Adrián
public void addValutak(String Valuta) {
    ContentValues values = new ContentValues();
    values.put("valuta", Valuta);
    db.insert( table: "Valutak",  nullColumnHack: null, values);
}

2 usages  Adrián
public void addUrmertekek(String Urmertek) {
    ContentValues values = new ContentValues();
    values.put("urmertek", Urmertek);
    db.insert( table: "Urmertekek",  nullColumnHack: null,values);
}

2 usages  Adrián
public void addTavolsagok(String Tavolsag) {
    ContentValues values = new ContentValues();
    values.put("tavolsag", Tavolsag);
    db.insert( table: "Tavolsagok",  nullColumnHack: null, values);
}

2 usages  Adrián
public void addUzemanyagok(String Megnev) {
    ContentValues values = new ContentValues();
    values.put("megnev", Megnev);
    db.insert( table: "Uzemanyagok", nullColumnHack: null,values);
}
```

- addTankolások() metódusban megadtam a szükséges bemeneti teszt adatokat, és azoknakmegadási sorrendjét.

```java
9 usages  Adrián
public void addTankolasok(long Datum, int AutoId, int Megtett_tav, int TavolsagId, float Ar, int ValutaId, float Menny, int UzemanyagId, int UrmertekId ) {
    ContentValues values = new ContentValues();
    values.put("datum", Datum);
    values.put("autoId", AutoId);
    values.put("megtett_tav", Megtett_tav);
    values.put("tavolsagId", TavolsagId);
    values.put("ar", Ar);
    values.put("valutaId", ValutaId);
    values.put("menny", Menny);
    values.put("uzemanyagId", UzemanyagId);
    values.put("urmertekId", UrmertekId);

    db.insert( table: "Tankolasok", nullColumnHack: null,values);

}
```

- feltolt() metódusban néhány próba tankolást adtam meg, fiktív tankolási adatokkal

```java
1 usage  Adrián
private void feltolt(){
    addAutok( Rendszam: "ABC-123",  megj: "szürke");
    addAutok( Rendszam: "DEF-456",  megj: "kék");

    addTankolasok(LocalDate.of( year: 2023, month: 4, dayOfMonth: 18).toEpochDay(),  AutoId: 2,  Megtett_tav: 150,  TavolsagId: 2,  Ar: 20,  ValutaId: 2,  Menny: 23, UzemanyagId: 1,  UrmertekId: 1);
    addTankolasok(LocalDate.of( year: 2023, month: 4, dayOfMonth: 2).toEpochDay(),  AutoId: 2,  Megtett_tav: 276,  TavolsagId: 1,  Ar: 2000,  ValutaId: 1,  Menny: 18, UzemanyagId: 1,  UrmertekId: 1);

    addTankolasok(LocalDate.of( year: 2023, month: 3, dayOfMonth: 11).toEpochDay(),  AutoId: 1,  Megtett_tav: 350,  TavolsagId: 1,  Ar: 2560,  ValutaId: 1,  Menny: 27, UzemanyagId: 2,  UrmertekId: 1)
    addTankolasok(LocalDate.of( year: 2023, month: 3, dayOfMonth: 26).toEpochDay(),  AutoId: 1,  Megtett_tav: 220,  TavolsagId: 2,  Ar: 50,  ValutaId: 2,  Menny: 10, UzemanyagId: 2,  UrmertekId: 2);
    addTankolasok(LocalDate.of( year: 2023, month: 2, dayOfMonth: 2).toEpochDay(),  AutoId: 1,  Megtett_tav: 220,  TavolsagId: 2,  Ar: 50,  ValutaId: 2,  Menny: 10, UzemanyagId: 2,  UrmertekId: 2);

    addTankolasok(LocalDate.of( year: 2022, month: 2, dayOfMonth: 2).toEpochDay(),  AutoId: 1,  Megtett_tav: 220,  TavolsagId: 2,  Ar: 50,  ValutaId: 2,  Menny: 10, UzemanyagId: 2,  UrmertekId: 2);
    addTankolasok(LocalDate.of( year: 2022, month: 2, dayOfMonth: 2).toEpochDay(),  AutoId: 1,  Megtett_tav: 220,  TavolsagId: 2,  Ar: 50,  ValutaId: 2,  Menny: 10, UzemanyagId: 2,  UrmertekId: 2);
    addTankolasok(LocalDate.of( year: 2022, month: 2, dayOfMonth: 2).toEpochDay(),  AutoId: 1,  Megtett_tav: 220,  TavolsagId: 2,  Ar: 50,  ValutaId: 2,  Menny: 10, UzemanyagId: 2,  UrmertekId: 2);
}
```

Végezetül meghívtam a „feltolt()” metódust a DatabaseHelper.java osztály „onCreate()” metódusában.

TA-13: Utolsó tankolás adatai, átlagfogyasztás, összes megtett út

KezdoFragment.java osztálynak az OnStart() metódusában dolgoztam

```
@Override
public void onStart() {
    super.onStart();
    MainActivity.getContext().showUjtankolasBtn();
    View view = getView();
    if (view != null) {
        //Új tankolás gomb navigáljon a felületre
        Button ujTankolasBtn = view.getRootView().findViewById(R.id.ujTankolasBtn);
        Button aktJarmuBtn = view.findViewById(R.id.aktJarmuBtn);

        TextView eltelt = (TextView) view.findViewById(R.id.eltelt_ido);
        TextView mennyiseg = (TextView) view.findViewById(R.id.uzemanyag_mennyiseg);
        TextView tipus = (TextView) view.findViewById(R.id.uzemanyag_tipus);
        TextView ar = (TextView) view.findViewById(R.id.uzemanyag_ar);
        TextView utan = (TextView) view.findViewById(R.id.km_utan);

        TextView atlag = (TextView) view.findViewById(R.id.atlagfogyasztas_text);
        TextView osszes = (TextView) view.findViewById(R.id.osszesut_text);

        aktJarmuBtn.setOnClickListener(Navigation.createNavigateOnClickListener(R.id.action_nav_kezdo_to_nav_jarmuvek));
        DatabaseHelper dbHelper = DatabaseHelper.getInstance(MainActivity.getContext());
```

- id alapján megkerestem a szöveges mezőket, amibe az adatok kerülnek majd
- egyenlővé tettem őket az adott TextView típusú változókkal, ezáltal tudunk rá hivatkozni egyszerűen

```
    //ha nincs jármű jelenítsük meg a megfelelő szöveget és legyen vége az onStart()-nak
    if(dbHelper.getJarmuvekSzama()==0){
        eltelt.setVisibility(View.GONE);
        mennyiseg.setVisibility(View.GONE);
        tipus.setVisibility(View.GONE);
        ar.setVisibility(View.GONE);
        utan.setVisibility(View.GONE);
        atlag.setVisibility(View.GONE);
        osszes.setVisibility(View.GONE);
        TextView osszesut = (TextView) view.findViewById(R.id.osszesut);
        osszesut.setVisibility(View.GONE);
        TextView atlagfogy = (TextView) view.findViewById(R.id.atlagfogyasztas);
        atlagfogy.setVisibility(View.GONE);
        TextView cim = view.findViewById(R.id.utolsoTankCimTxt);
        cim.setVisibility(View.GONE);
        ujTankolasBtn.setVisibility(View.GONE);

        binding.nincsAutoTxtKezdo.setVisibility(View.VISIBLE);
        aktJarmuBtn.setText("Járművek oldal");
        return;
    }
```

Ha nincs megadva jármű az alkalmazásban, akkor az adott, ehhez tartozó figyelemfelhívó szövegmellett beállítottam a nézetek View-ok láthatóságát is. Ebben az esetben nem láthatóak, ezzel biztosítva, hogy a felhasználó csak ténylegesen megadott jármű esetében tudja használni az alkalmazás lényeges elemeit.

```
//ha nincs tankolás jelenítsük meg a megfelelő szöveget és legyen vége az onStart()-nak
if(dbHelper.getTankolasokSzama()==0){
    eltelt.setVisibility(View.GONE);
    mennyiseg.setVisibility(View.GONE);
    tipus.setVisibility(View.GONE);
    ar.setVisibility(View.GONE);
    utan.setVisibility(View.GONE);
    atlag.setVisibility(View.GONE);
    osszes.setVisibility(View.GONE);
    TextView osszesut = (TextView) view.findViewById(R.id.osszesut);
    osszesut.setVisibility(View.GONE);
    TextView atlagfogy = (TextView) view.findViewById(R.id.atlagfogyasztas);
    atlagfogy.setVisibility(View.GONE);
    TextView cim = view.findViewById(R.id.utolsoTankCimTxt);
    cim.setVisibility(View.GONE);

    TextView nincsTank = view.findViewById(R.id.nincsTankTxt);
    nincsTank.setVisibility(View.VISIBLE);
    return;
}
```

Hiányzó tankolás esetén az előző esethez hasonlóan, itt is a láthatóságot korlátozzuk, és hívjuk fel afigyelmet egy új tankolást rögzítésére.

```
TankolasOsszetett tankolasOsszetett = dbHelper.getTankolasokByAutoId(aktivJarmu.getAutoId()).get(0);
Stat stat = new Stat();
String megtettUt = tankolasOsszetett.xMegtettUt();
String tankoltMennyiseg = tankolasOsszetett.xTankoltMennyiseg();
String uzemanyag = tankolasOsszetett.getUzemanyag();
LocalDate datum = tankolasOsszetett.getDatum();
String egysegar = tankolasOsszetett.xEgysegar();
float atlagFogy = stat.atlagFogy100kmen();
float osszesUt = stat.osszesMegtettKm();

eltelt.setText(DAYS.between(datum, LocalDate.now())+" napja");
mennyiseg.setText(tankoltMennyiseg);
tipus.setText(uzemanyag);
ar.setText(egysegar);
utan.setText(megtettUt+" után");

DecimalFormat df = new DecimalFormat( pattern: "#.####");
atlag.setText(df.format(atlagFogy)+ " l/100km");
osszes.setText(osszesUt + " km");
```

- Az adatbázisban szerepelt ID alapján tudunk a járművekre, tankolási adatokra szűrni
- Meghívtam a TankolasOsszetett.java osztályban lévő metódusokat
- A megfelelő helyre kiírattam a metódusok által visszaadott értékeket

TA-25: Eddigi hardcoded string-ek átszervezése string.xml-be

➔ layout package

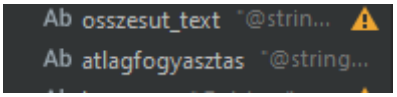
Az adott felületen lévő elemek kézzel megadott Text értékeinek a módosítását hajtottam végre, melyek hozzáadódtak a string.xml-hez

Például fragment_kezdo.xml felületi elemeinek megadása kóddal:

```
<TextView
    android:id="@+id/osszesut_text"
    android:layout_width="377dp"
    android:layout_height="30dp"
    android:layout_marginStart="20dp"
    android:layout_marginTop="10dp"
    android:layout_marginEnd="20dp"
    android:text="Km"
    android:textAlignment="viewEnd"
    android:textColor="@color/tankolas_felvetel_doboz"
    android:textSize="20sp"
    android:textStyle="bold"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.666"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/osszesut" />


<TextView
    android:id="@+id/atlagfogyasztas"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="20dp"
    android:layout_marginTop="10dp"
    android:text="Átlagfogyasztás:"
    android:textColor="@color/design_default_color_background"
    android:textSize="18sp"
    android:textStyle="bold"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/km_utan" />
```

És a Design módban látva:



TA-32: Ne lehessen jövőbeli dátumú tankolást felvenni:

TankolasFelvetelFragment.java

Az osztály OnClick() metódusában be lett állítva, hogy a tankolás megadásának aktuális napjánálkésőbbi napot ne lehessen megjelölni, mint időpontot.

Szimplán nem lehet kijelölni a jelenlegi napnál későbbi dátumot.

```
    @Override
    public void onClick(View v) {
        Calendar calendar = Calendar.getInstance();
        int year = calendar.get(Calendar.YEAR);
        int month = calendar.get(Calendar.MONTH);
        int day = calendar.get(Calendar.DAY_OF_MONTH);
        /**
         * Új dátumválasztó párbeszédpanelt hoz létre a megadott dátumhoz.
         */
        DatePickerDialog dialog = new DatePickerDialog(getContext(), android.R.style.Theme_DeviceDefault_Dialog, dateSetListener,year,month,day);
        dialog.show();
        dialog.getDatePicker().setMaxDate(new Date().getTime());
        /**
         * a jelenlegi napnál későbbi dátumok kiválasztásának "tiltása"
         * */
    }
});
```