Mobil programozási alapok

Költekezés nyilvántartó mobilalkalmazás

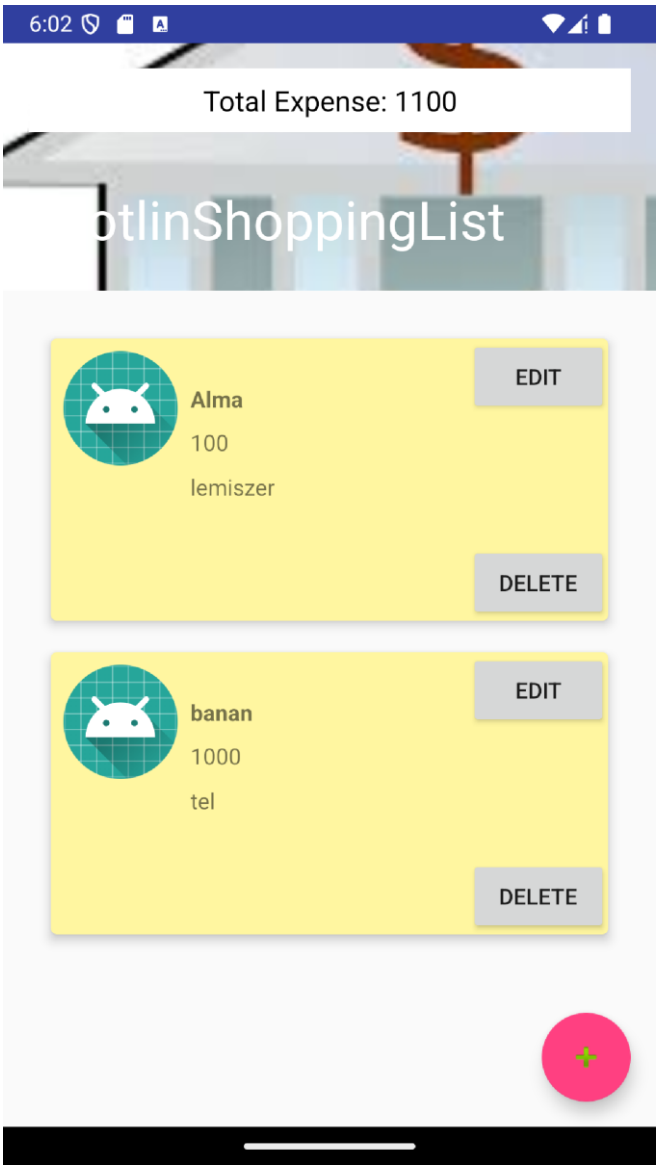
Dobai Attila

(DIZ4VX)

Tartalomjegyzék

Felhasználói felület:

Költekezés nyilvántartó mobilalkalmazás egy egyszerű, helyi adatbázison alapuló költekezés menedzsmentre alkalmas app. A felhasználó létre tud hozni újköltekezéseket és szerkeszteni, végül törölheti is őket. Az app felül kiírja az eddig elköltött összeget.

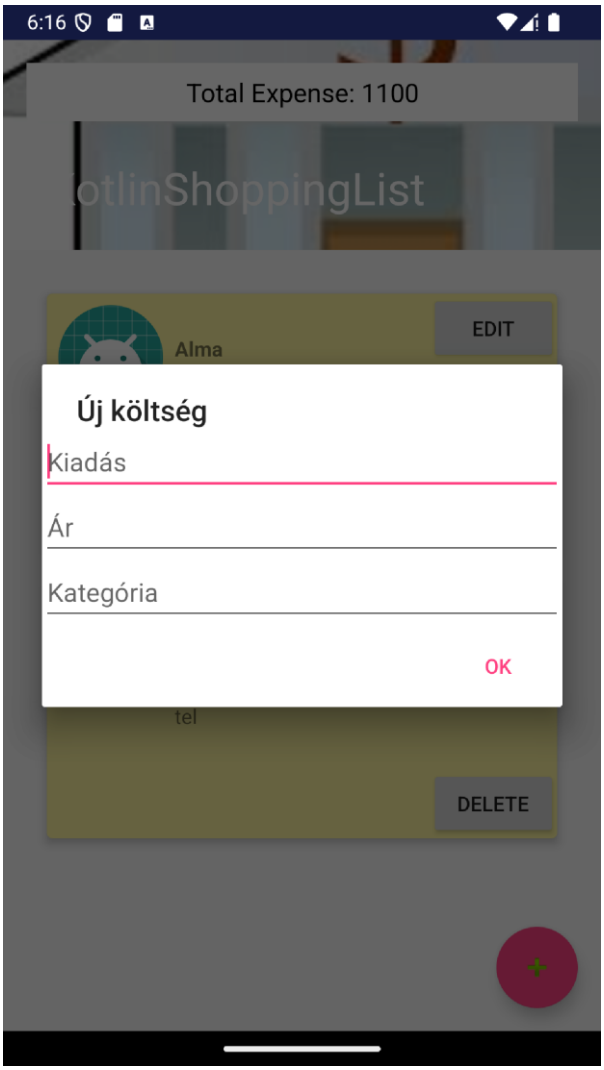


A feladatok listázása:

Az alkalmazás a kezdő képernyő tetején kiírja az eddigi költségeket. Alatta helyezkednek el a költségek tételesen. A jobb alsó sarokban a gombra kattintva lehet felvenni újabb költséget.

A feladatok felvitele és módosítása:

Az egyes költekezések egy listában jelennek meg kártyák formájában. A kártyákon megjelenő gombok segítségével módosíthatóak és törölhetőek. A feladatok módosításakor a létrehozásnál is megjelenő felugró ablakban módosíthatunk minden megadott értéket.



A programkód:

Az alkalmazás az Android Studio Bumblebee verziójában Kotlin nyelven történt leprogramozásra. Az alkalmazás forrásfile-jai az alábbi github repositoryban elérhetőek:

Az adatbázis:

Az alkalmazás alapját adó adatbáziskapcsolatot az AppDatabase osztály menedzseli, melynek egyetlen példánya a getInstance() függvény meghívásával kérhető le, illetve ennek legelső meghívásakor jön létre. Az adatok egy lokális expenses.db nevű file-ba kerülnek mentésre.

```
abstract class AppDatabase : RoomDatabase() {

    abstract fun expenseDao(): ExpenseDAO

    companion object {
        private var INSTANCE: AppDatabase? = null

        fun getInstance(context: Context): AppDatabase {
            if (INSTANCE == null) {
                INSTANCE = Room.databaseBuilder(
                    context.applicationContext,
                    AppDatabase::class.java,  name: "expenses.db"
                ).build()
            }
            return INSTANCE!!
        }

        fun destroyInstance() {
            INSTANCE = null
        }
    }
}
```

Expense egyed:

Az egyedek definíciója a Expense modell osztályban került implementálásra. Az egyed a felhasználói felületen is megjelenő adatokat tartalmazza. Ez alól kivételt képez az elsődleges kulcs, mely az adatbázis által generált egyedi azonosító. Ezt az adatbázis az egyed mentésekor generálja.

```
@Entity(tableName = "expenses")
data class Expense(
    @PrimaryKey(autoGenerate = true) var expenseId: Long?,
    @ColumnInfo(name = "description") var description: String,
    @ColumnInfo(name = "amount") var amount: Int,
    @ColumnInfo(name = "category") var category: String
) : Serializable
```

ExpenseDAO:

A ExpenseDAO interface felelős az adatokon végezhető adatbázisműveletek definiálására. Itt az összes feladat lekérdezése, új létrehozása, illetve adott feladat módosítása és törlése szerepel. Ezen felül itt kerül lekérdezésre az eddig összesen elköltött összeg.

```kotlin
@Dao
interface ExpenseDAO {

    // Az összes költség listázása
    @Query( value: "SELECT * FROM expenses")
    fun findAllExpenses(): List<Expense>

    // Egy költség beszúrása
    @Insert
    fun insertExpense(expense: Expense): Long

    // Egy költség törlése
    @Delete
    fun deleteExpense(expense: Expense)

    // Egy költség módosítása
    @Update
    fun updateExpense(expense: Expense)

    @Query( value: "SELECT SUM(amount) FROM expenses")
    fun getTotalExpense(): Int
}
```

Main Activity:

A Main Activity file-ban található függvények vezérlik az alkalmazás fő futását. Itt kerültek implementálásra az alkalmazás indításakor és leállításakor kezelendő akciók, valamint a feladatok listájának létrehozása és azok kezelése. Egy feladat módosítása vagy egy új létrehozása esetén a ExpenseDialog osztály kerül meghívásra.

ExpenseDialog:

Ezen osztály kezeli azon dialógust mely a létrehozáskor és módosításkor ugrik fel. Itt kerülnek az értékek módosításra vagy megadásra.

ExpenseAdapter:

Végül a ExpenseAdapter osztály felel a feladatokat listázó RecyclerView tartalmának kezeléséért. Az items lista tartalmazza az alkalmazás használata közben folyamatosan változó expense objektumokat, melyeket a létrehozás szerint rendez sorba. Az osztály szorosan együttműködik a row_item.xml fileal mivel ennek segítségével hozza létre és menedzseli a feladat kártyákat az alkalmazás.