

Tokaj-Hegyalja Egyetem

Mobilprogramozás (THE_00585_N_5_G) F1 Mini Shop

Project leírása



Projekt leírása	4
Főbb funkciók	5
Projekt felépítése	6
AndroidManifest.xml	7
MainActivity.kt	8
CartActivity.kt	9
CartManager.kt	10
Product.kt	11
CartItem.kt	12
Problémák	13
Főbb függőségek build grande	14
Termékek	15
Tesztelések	16
Design elemek	17
Megjegyzések	18

Projekt leírása

- **Projekt neve:** MiniShop1
- **Platform:** Android (Kotlin)
- **Fejlesztés dátuma:** 2025. december 04-05.
- **Célja:** Könnyen érthető, oktatási célú webshop alkalmazás létrehozása, amely bemutatja a terméklista-kezelés, kosárlogika és rendelési folyamat működését.
- **Fő funkciók:**
 - Termékek listázása képekkel, árakkal és leírással
 - Termékadatlap megnyitása és kosárba helyezés
 - Dinamikus kosárkezelés (hozzáadás, mennyiség csökkentése/növelése, törlés)
 - Végösszeg automatikus számítása
 - Rendelési adatok megadása (név, cím, fizetési mód)
 - Rendelés véglegesítése visszaigazoló párbeszédablakkal
- **Felhasznált technológiák:**
 - Kotlin nyelv
 - Android View alapú UI
 - Layout inflating termék- és kosárelemekhez
 - AlertDialog és Snackbar értesítések
 - Saját modellréteg (Product, CartItem, CartManager)
- **Architektúra:**
 - Külön adatmodellek a termékekhez és kosártételekhez
 - Egy központi kosárkezelő (CartManager) a mennyiségek és műveletek kezelésére
 - Két fő képernyő: terméklista (MainActivity) és kosár oldal (CartActivity)
- **Használati cél:**
 - Alkalmas tanuláshoz, Android-fejlesztés gyakorlásához
 - Alap webshop funkciók demonstrálása backend nélkül
 - Később bővíthető REST-es háttérrel, felhasználói fiókokkal, valós fizetéssel

Főbb funkciók

1. Termékkezelés

- Előre definiált termékek listázása képpel, névvel, árral és leírással
- Termék részleteinek megjelenítése felugró ablakban
- Termék kosárba helyezése egy kattintással

2. Kosárkezelés

- Kosár tartalmának dinamikus megjelenítése
- Mennyiség növelése/csökkentése (+ / – gombokkal)
- Termékek eltávolítása a kosárból
- Kosár teljes értékének automatikus újraszámítása

3. Rendelési folyamat

- Vevő adatok bekérése (név, szállítási cím, e-mail cím, telefonszám)
- Fizetési mód kiválasztása legördülő listából
- Adatellenőrzés (kötelező mezők, fizetési mód kiválasztása)
- Sikeres rendelés visszaigazolása párbeszédablakban
- Kosár automatikus ürítése rendelés után

4. Felhasználói élmény és visszajelzések

- Visszajelzés Snackbar-kal termék kosárba helyezésekor
- Hibajelzések (üres mezők, üres kosár)
- Egyszerű, áttekinthető kétképernyős felépítés (terméklista + kosár)

5. Technikai megvalósítás

- Külön adatmodellek (Product, CartItem) és központi kosárkezelő (CartManager)
- Dinamikus nézetinflálás termék- és kosárelemekhez
- Külön Activity a főoldalnak (MainActivity) és a kosárnak (CartActivity)

Projekt felépítése

1. Kotlin fájlok (.kt)

MainActivity.kt

- Főképernyő, termékek listázása
- Termékek dinamikus felfűzése egy vertikális LinearLayout-ba
- Termék részleteinek megjelenítése egy AlertDialog-ban
- „Kosárba” gomb: termék hozzáadása a kosárhoz (CartManager)
- Visszajelzés Snackbar segítségével
- „Kosár” gomb: navigáció a CartActivity-hez

CartActivity.kt

- Kosárban lévő tételek megjelenítése dinamikusán
- Mennyiség módosítása (+ / – gombokkal)
- Termék eltávolítása a kosárból
- Összár automatikus kalkulációja és kiírása
- Vevőadatok bekérése (név, cím)
- Fizetési mód kiválasztása Spinner segítségével
- „Rendelés” gomb: adatok ellenőrzése, visszaigazoló párbeszédablak, kosár ürítése

CartManager.kt

- Kosár központi kezelése (hozzáadás, növelés, csökkentés, törlés)
- Kosár teljes értékének kiszámítása
- Kosár ürítése rendelés után
- Product.kt
- Egy termék adatszerkezete
- Mezők: azonosító, név, leírás, ár (Ft), kép erőforrásazonosító

CartItem.kt

- Egy kosár tétel modellje
- Mezők: Product példány + mennyiség
- Alapértelmezett mennyiség: 1

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android">

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.MiniShop">

        <activity android:name=".CartActivity" />

        <activity
            android:name=".MainActivity"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>

    </application>

</manifest>
```

MainActivity.kt

Főképernyő, termékek listázása és kosárhoz adás.

MainActivity

- productsContainer: LinearLayout – a termékkártyák szülőnézete
- products: List<Product> – előre definiált termékek listája (sapka, póló, bögre)
- onCreate(savedInstanceState)
 - Layout beállítása (activity_main)
 - ActionBar elrejtése
 - productsContainer és btnOpenCart bekötése
 - inflateProducts() meghívása
 - „Kosár” gomb → CartActivity indítása Intent-tel
- inflateProducts()
 - item_product layout felfűzése minden termékhez
 - Kép, név, ár beállítása (ivProductImage, tvProductName, tvProductPrice)
 - „Részletek” gomb → showProductDetailDialog(product) hívása
 - Nézet hozzáadása a productsContainer-hez
- showProductDetailDialog(product: Product)
 - dialog_product_detail layout felfűzése
 - Részletes név, leírás, ár, kép kitöltése
 - „Kosárba” gomb:
 - CartManager.addProduct(product) – termék kosárhoz adása
 - Snackbar megjelenítése: visszajelzés a felhasználónak
 - Dialógus bezárása

CartActivity.kt

Kosár képernyő, mennyiségkezelés, rendelés leadása.

CartActivity

- cartItemsContainer: LinearLayout – kosárban lévő tételek listanézete
- tvTotal: TextView – teljes ár kijelzése
- etName: EditText – vevő neve
- etAddress: EditText – szállítási cím
- spinnerPayment: Spinner – fizetési mód kiválasztása
- btnPlaceOrder: Button – rendelés véglegesítése
- btnBackToShop: Button – visszalépés a főoldalra
- onCreate(savedInstanceState)
 - Layout beállítása (activity_cart)
 - Nézetek bekötése findViewById-dal
 - setupPaymentSpinner() – fizetési módok feltöltése
 - loadCart() – kosár tartalmának betöltése
 - „Rendelés” gomb → placeOrder()
 - „Vissza a boltba” gomb → finish() (Activity bezárása)
- onResume()
 - loadCart() újbóli meghívása, hogy a kosár mindig friss legyen
- setupPaymentSpinner()
 - Fizetési módok listája: „Válassz...”, „Utánvét”, „Bankkártya”, „Átutalás”
 - ArrayAdapter létrehozása és hozzárendelése a Spinner-hez
- loadCart()
 - cartItemsContainer kiürítése
 - Végigiterál a CartManager.items listán
 - Minden CartItem-hez: item_cart layout felfűzése
 - Kép, név, egységár, mennyiség beállítása
 - btnMinus: CartManager.decreaseQuantity(item.product) → újratöltés
 - btnPlus: CartManager.increaseQuantity(item.product) → újratöltés
 - btnRemove: CartManager.removeProduct(item.product) → újratöltés
 - CartManager.getTotal() → „Összesen: X Ft” kiírása a tvTotal-ba
- placeOrder()
 - Név, cím, fizetési mód beolvasása
 - Ellenőrzések:
 - Üres kosár esetén Toast („A kosár üres.”)
 - Üres név/cím mezők esetén hiba (etName.error, etAddress.error)
 - Nincs fizetési mód kiválasztva → Toast
 - Siker esetén AlertDialog megjelenítése „Rendelés sikeres” üzenettel
 - „OK” gomb:
 - CartManager.clearCart() – kosár törlése
 - loadCart() – üres kosár megjelenítése
 - Név, cím, fizetési mód alaphelyzetbe állítása
 - finish() – visszalépés a főoldalra

CartManager.kt

Központi kosárkezelő objektum.

CartManager

- items: MutableList<CartItem> – kosár tartalma (termék + mennyiség)
- addProduct(product: Product)
 - Megkeresi, hogy a termék már szerepel-e a kosárban
 - Ha igen: quantity++
 - Ha nem: új CartItem(product, 1) hozzáadása
- increaseQuantity(product: Product)
 - Megkeresi a hozzá tartozó CartItem-et
 - quantity++
- decreaseQuantity(product: Product)
 - Megkeresi a hozzá tartozó CartItem-et
 - quantity--
 - Ha az új mennyiség $\leq 0 \rightarrow$ elem eltávolítása a listából
- removeProduct(product: Product)
 - Minden, adott termékhez tartozó CartItem eltávolítása
- clearCart()
 - Teljes kosár kiürítése (items.clear())
- getTotal(): Int
 - Összár kiszámítása: $\text{sumOf} \{ \text{it.product.price} * \text{it.quantity} \}$

Product.kt

Termék adatmodell.

Product

- data class Product
 - id: Int – egyedi azonosító
 - name: String – termék neve
 - description: String – termék leírása
 - price: Int – ár forintban
 - imageResId: Int – drawable erőforrás ID (termékkép)

CartItem.kt

Kosár tétel adatmodell.

CartItem

- data class CartItem
 - product: Product – a kosárba helyezett termék
 - quantity: Int = 1 – mennyiség (alapértelmezés: 1 darab)

Problémák

Kosár üresen rendeléskor crash

- **Probléma:** Üres kosár esetén a „Rendelés” gomb megnyomásakor a kód azonnal a megrendelési folyamatot futtatta, ami később logikai hibákat és váratlan viselkedést okozott.
- **Megoldás:** A placeOrder() elejére bekerült egy ellenőrzés, amely üres kosár esetén Toast üzenetet jelenít meg, és leállítja a metódust:
- ```
if (CartManager.items.isEmpty()) {
```
- ```
    Toast.makeText(this, "A kosár üres.", Toast.LENGTH_SHORT).show()
```
- ```
 return
```
- ```
}
```

Nem frissült a mennyiség a + / – gomb megnyomása után

- **Probléma:** A mennyiséget módosító gombok (btnPlus, btnMinus) csak a modellben (CartManager) változtatták az értéket, de a felületen a darabszám és az összár nem frissült azonnal.
- **Megoldás:** Minden mennyiségmódosító művelet után meghívásra kerül a loadCart(), amely újrarajzolja a kosár elemeit és újraszámolja az összeget:
- ```
btnMinus.setOnClickListener {
```
- ```
    CartManager.decreaseQuantity(item.product)
```
- ```
 loadCart()
```
- ```
}
```
- ```
btnPlus.setOnClickListener {
```
- ```
    CartManager.increaseQuantity(item.product)
```
- ```
 loadCart()
```
- ```
}
```

Kosár kiürítése után régi adatok maradtak a képernyőn

- **Probléma:** A sikeres rendelés után a kosár tartalmát ugyan törölte a CartManager.clearCart(), de a felhasználói felület (lista és összár) nem frissült, így úgy tűnt, mintha még lennének tételek.
 - **Megoldás:** A kosár törlése után ismét meghívásra kerül a loadCart(), ami kiüríti a listanézetet és 0 Ft-ra állítja az összeget:
 - ```
CartManager.clearCart()
```
- loadCart()

## Főbb függőségek build grande

```
dependencies {
 // Kotlin alapkönyvtár
 implementation "org.jetbrains.kotlin:kotlin-stdlib:1.9.0"

 // AppCompat – AppCompatActivity, AlertDialog
 implementation "androidx.appcompat:appcompat:1.6.1"

 // Material Components – MaterialButton, Snackbar
 implementation "com.google.android.material:material:1.11.0"

 // Core KTX – Kotlin kiterjesztések
 implementation "androidx.core:core-ktx:1.12.0"
}
```

## Termékek

### 1. F1 Sapka

- **ID:** 1
- **Név:** F1 Sapka
- **Leírás:** Hivatalos Forma-1 rajongói sapka.
- **Ár:** 7 990 Ft
- **Kép:** ic\_product\_cap

### 2. F1 Póló

- **ID:** 2
- **Név:** F1 Póló
- **Leírás:** Kényelmes F1 póló, több méretben.
- **Ár:** 9 990 Ft
- **Kép:** ic\_product\_tshirt

### 3. F1 Bögre

- **ID:** 3
- **Név:** F1 Bögre
- **Leírás:** Kerámia bögre F1 logóval.
- **Ár:** 3 990 Ft
- **Kép:** ic\_product\_mug

## **Tesztelések**

- Alkalmazás elindul
- Termékek megjelennek képpel és árral
- Termék részletek megnyithatók
- Kosárba rakás működik
- Mennyiség növelése/csökkentése működik
- Termék törlése a kosárból működik
- Összár minden művelet után frissül
- Fizetési oldal elemei megjelennek (adatmezők + fizetési mód)
- Adatvalidáció működik (kötelező mezők, fizetési mód)
- Rendelés sikeresen véglegesíthető
- Rendelés után a kosár automatikusan kiürül

## Design elemek

- **Színvilág:**
  - Fő gombok: lila / magenta árnyalatok
  - Visszajelzések (Snackbar): Material alap színek
  - Összár kiemelése: zöld szöveg
- **Tipográfia:**
  - Címek félkövéren
  - Szövegméret: 14–24sp tartományban
  - Terméknevek kiemelve, árak jól olvasható méretben
- **Gombok:**
  - Lekerekített sarkok
  - Egységes belső margó (padding)
  - Minimum magasság beállítás az érintési felület miatt
  - Kontrasztos színek a jobb olvashatóságért
- **Mennyiségválasztó:**
  - Kis méretű + / – gombok (kb. 32×32dp)
  - Középen a mennyiség száma
  - Jól elkülönülő háttérszín a kosárelem dobozában
- **Layout felépítése:**
  - Többnyire LinearLayout alapú szerkezet
  - Függőleges görgetés támogatása ScrollView-val hosszú kosár esetén
  - LayoutInflater használata dinamikus elemekhez (termékek, kosár tételek)



## **Megjegyzések**

- Az alkalmazás teljes mértékben lokálisan fut, nem kapcsolódik külső szerverhez vagy adatbázishoz.
- A rendelések nincsenek elmentve, a kosár tartalma csak futásidőben él.
- A fizetés kizárólag szimulált folyamat, valós tranzakció vagy fizetési rendszer integrációja nem történik.
- A termékek statikus erőforrások, a drawable mappából kerülnek betöltésre.
- A kosárlogika memóriaalapú (CartManager), az alkalmazás újraindításakor törlődik.
- A navigáció két fő Activity között történik, fragmenteket jelenleg nem használ.
- A layoutok egyszerűsített felépítésűek, a funkcionalitás demonstrálását szolgálják.