

Semestrálna práca z predmetu *Vývoj aplikácií pre mobilné zariadenia*

Backpack manager

**vypracoval:** Benjamín Sakala

**študijná skupina:** 5ZYI22

**program**: Informatika

**cvičiaci:** doc. Ing. Patrik Hrkút, PhD. **Školský rok:** 2023/2024

**termín cvičenia:** Pondelok 8:00 v Žiline dňa 5. 6. 2024

Obsah

[1. Popis aplikácie 3](#_Toc168590101)

[2. Podobné aplikácie 3](#_Toc168590102)

[2.1. Packing List Travel Packlist 3](#_Toc168590103)

[2.2. Packingbird – Packing List App 3](#_Toc168590104)

[3. Použitie aplikácie 4](#_Toc168590105)

[4. Návrh aplikácie 5](#_Toc168590106)

[4.1. Diagram prechodov medzi obrazovkami 5](#_Toc168590107)

[4.2. Obrazovka s predmetmi 6](#_Toc168590108)

[4.3. Obrazovka s predmetmi v batohu 6](#_Toc168590109)

[4.4. Obrazovka skupín 7](#_Toc168590110)

[4.5. Editovacia obrazovka 7](#_Toc168590111)

[4.6. Informačná obrazovka skupiny 8](#_Toc168590112)

[4.7. Obrazovka typov 8](#_Toc168590113)

[5. Implementácia 9](#_Toc168590114)

[5.1. Room databáza 9](#_Toc168590115)

[5.2. Navigation 9](#_Toc168590116)

[5.3. ViewModel 9](#_Toc168590117)

[5.4. LifeCycles 10](#_Toc168590118)

[5.5. Práca s lokálnym úložiskom aplikácie 10](#_Toc168590119)

[5.6. Získavanie obrázkov zo zariadenia 11](#_Toc168590120)

[6. Poznámky 12](#_Toc168590121)

[7. Zdroje 12](#_Toc168590122)

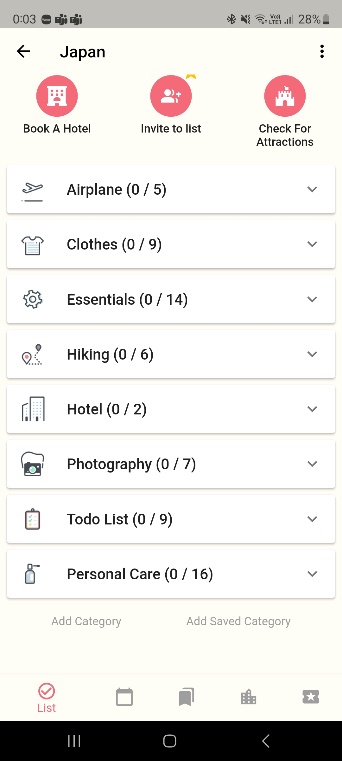
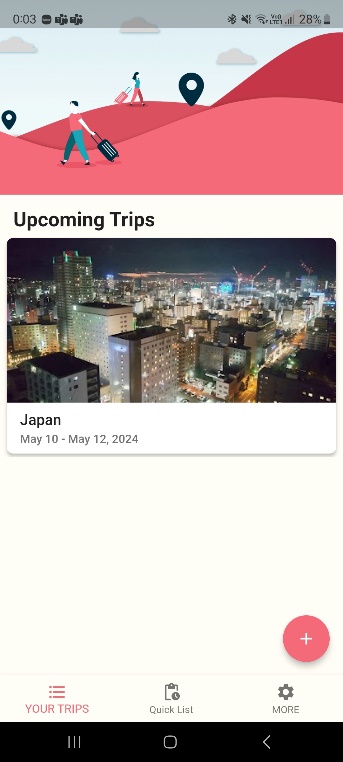
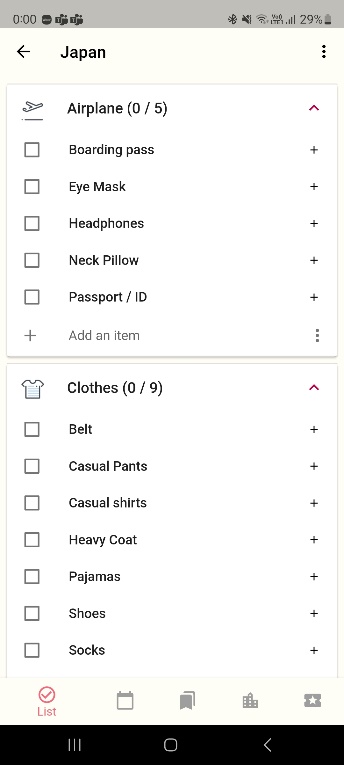
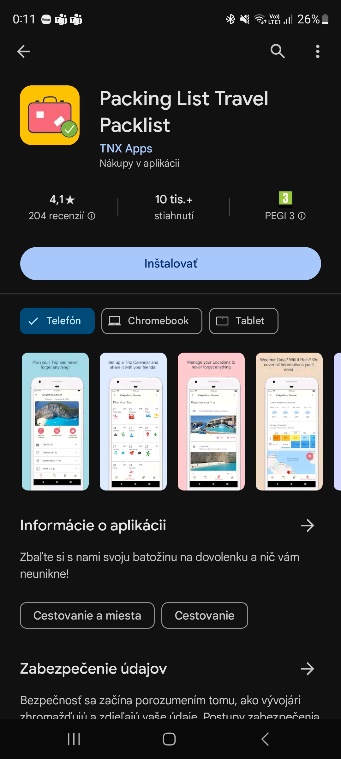
# Popis aplikácie

Aplikácia Backpack manager slúži na uchovávanie informácií a fotografií predmetov, ktoré si používateľ najčastejšie balí do ruksakov alebo kufrov. Takisto umožňuje, ukladať sady vecí, ktoré často berie so sebou na pracovné cesty, výlety alebo turistiky. Backpack manager dokáže vypočítať, koľko % celkovej váhy zaberajú jednotlivé kategórie predmetov, takisto aj váhu jednotlivých predmetov. Používateľ si môže vytvárať ľubovoľné kategórie typov predmetu do ktorých môže jeho predmety zaradzovať. Aplikácia si medzi jednotlivými spusteniami uchováva stav batohu a vďaka tomu nemusí používateľ vždy ukladať zoznamy do skupín.

# Podobné aplikácie

## Packing List Travel Packlist

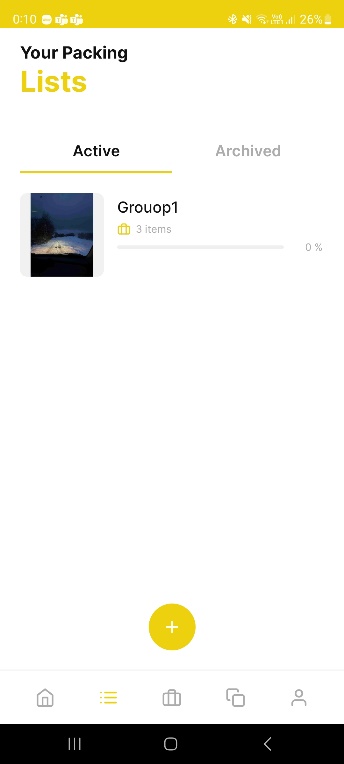
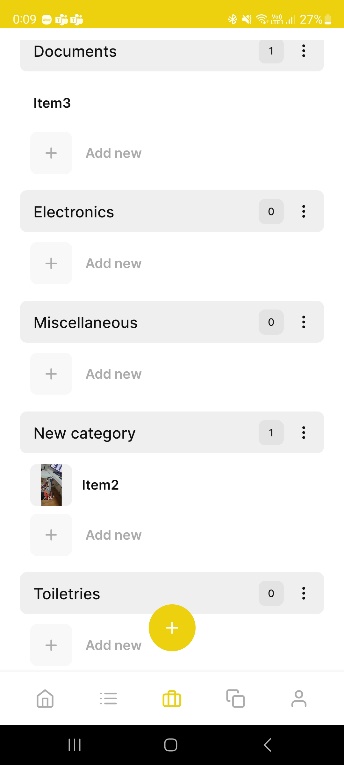
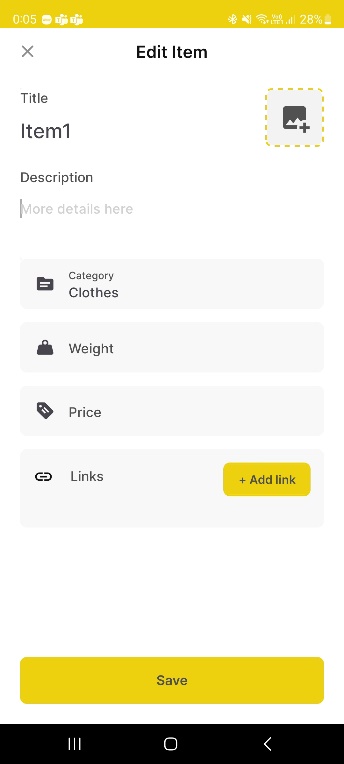
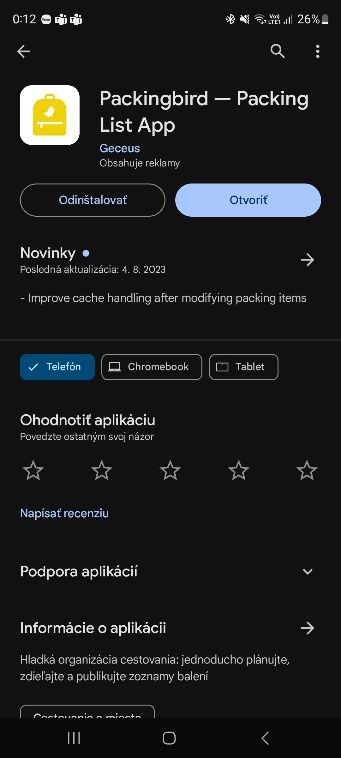
Táto aplikácia sa čiastočne podobá na Backpack manager svojou podstatou a to chystaní produktov na cesty. Funguje však iným spôsobom a líši sa aj inými funkciami. Pracuje skôr viac ako todo list. Váš list vytvorí podľa toho čo zadáte, automaticky (kam idete, na ako dlho, čo tam idete robiť). Jednotlivé položky sú zoradené do kategórií, ktoré fungujú ako osobitné check listy. Neumožňuje žiadnu správu váhy predmetov, ani ich obrázkov. K vašej ceste vám ponúkne rôzne aktivity, hotely a dokonca vám zistí počasie.



*Ukážka aplikácie*

## Packingbird – Packing List App

Packingbird je aplikácia podobná Backpack manager, funkcie má skoro rovnaké. Jednou z jej nevýhod sú však reklamy, ktoré používateľovi znepríjemňujú narábanie s aplikáciou. Výhody sú v založení zoznamu, ktorý funguje aj ako check list.



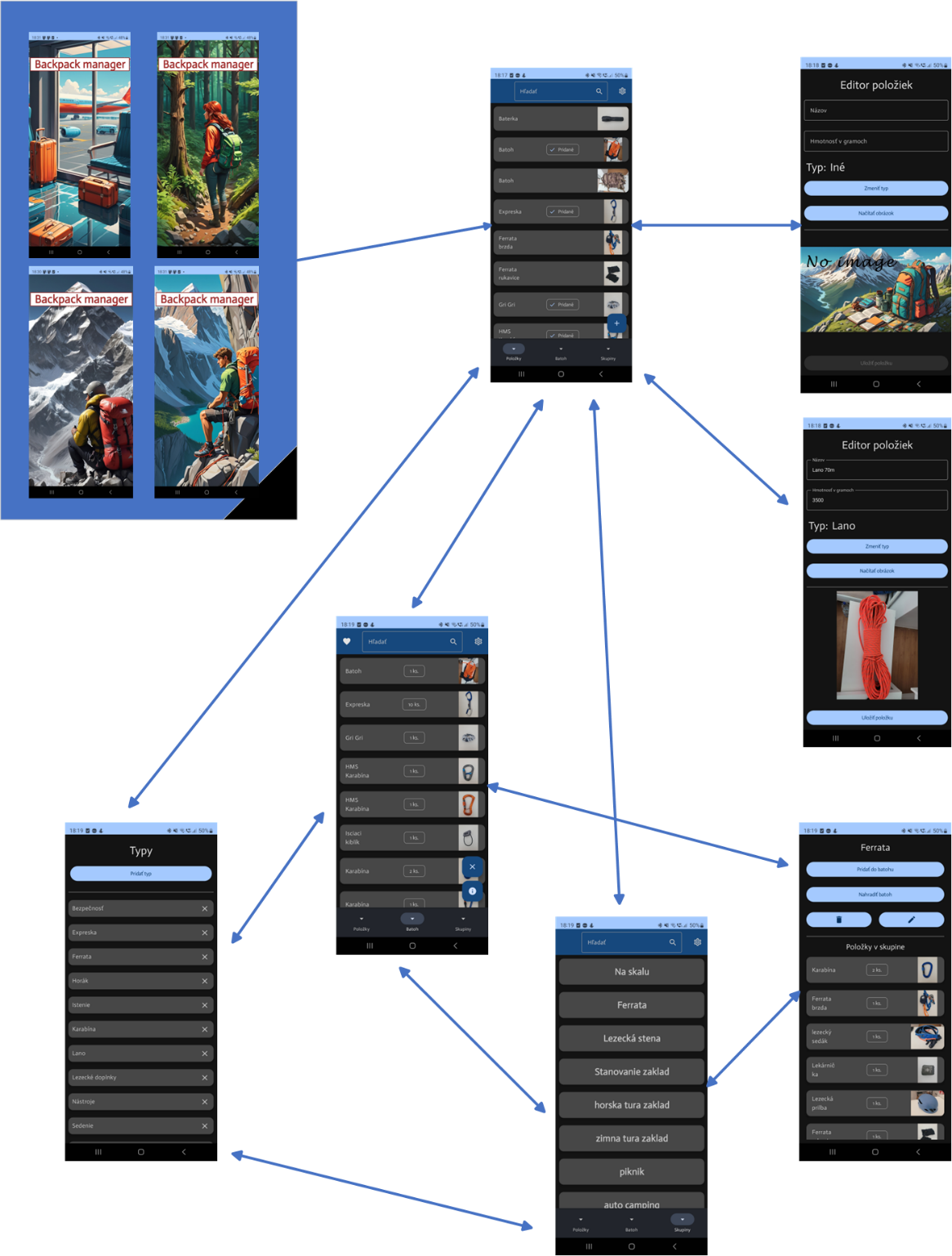
*Ukážka aplikácie*

# Použitie aplikácie

Backpack manager je úžasný pomocník pri chystaní vecí na túru, dovolenky, služobné cesty a veľa ďalších aktivít, kde si používateľ berie so sebou väčší počet vecí a je potrebné dávať pozor na váhu, napríklad turistika alebo preprava lietadlom. Rozpočítavanie váhy na jednotlivé kategórie vám umožňuje pomôcť pri rozhodovaní, ktoré z vecí si zbalíte a nezbalíte na cestu. Užitočnou vlastnosťou je aj ukladanie zoznamov, vďaka čomu si používatelia môžu pri opakovaných cestách znova nachystať rovnaké veci alebo skontrolovať, či nič nezabudli pred odchodom.

# Návrh aplikácie

## Diagram prechodov medzi obrazovkami

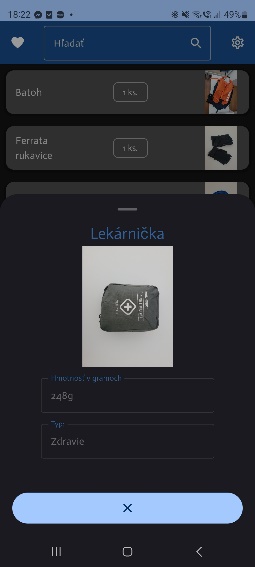


Aplikácia po spustení vyberá náhodnú uvítaciu obrazovku z 8 rôznych obrazoviek v diagrame sú naznačené ako súbor rôznych obrazoviek. Pod obrazovky ktorými sú rôzne informačné vysúvacie lišty boli z diagramu vynechané pretože sú súčasťami hlavných obrazoviek. Z obrazovky s predmetmi je editovacia obrazovka naznačená dva krát z dôvodu že sa dá otvoriť ako nový predmet ale aj ako úprava už existujúceho predmetu.

## Obrazovka s predmetmi

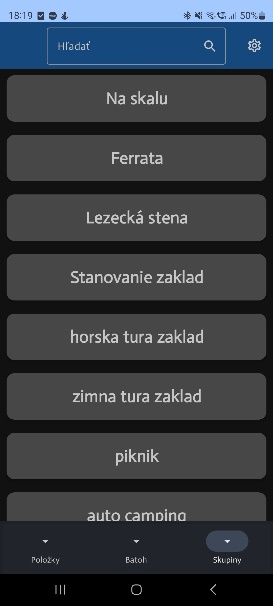
Obrazovka s predmetmi je považovaná ako hlavná obrazovka aplikácie. Je možné na nej pridať nový predmet stlačením tlačidla vpravo naboku zo symbolom +. Ak sa v zozname predmetov nachádzajú nejaké predmety je možné ich vyhľadávať. Pomocou textového poľa navrchu. Vyhľadávanie hľadá výsledky ktorých meno alebo typ obsahuje reťazec ktorý bol zadaný do vyhľadávanie. Vyhľadávanie nie je case sensitive. Po stlačení na ľubovoľnú kartu s predmetom sa zobrazí vysúvacia obrazovka s bližšími informáciami o predmete a možnosťami práce s ním. Daný predmet sa dá vymazať z aplikácie tlačidlom s ikonou koša. Po stlačení sa zobrazí vymazávacie dialógové okno. Pomocou tlačidla s symbolom pera sa otvorí editovacia obrazovka daného predmetu. Tlačidlom s označením + sa pridáva predmet do batohu. Po jeho stlačení sa zobrazí dialóg ktorý sa pýta na počet predmetov ktorý chce používateľ pridať do batohu. Je možné zadať akékoľvek celé číslo väčšie ako 0. Posledné tlačilo je viditeľné iba ak je už predmet v batohu a pomocou neho sa tento predmet z batohu odstráni. Poslednú časťou tejto obrazovky je tlačidlo vpravo hore zo symbolom ozubeného koliečka, toto tlačidlo otvorí obrazovku so pomocou ktorej sa vytvárajú a odstraňujú typy predmetov. Respektíve kategórie. Pomocou dolnej lišty sa dá navigovať medzi troma hlavnými obrazovkami.

## Obrazovka s predmetmi v batohu

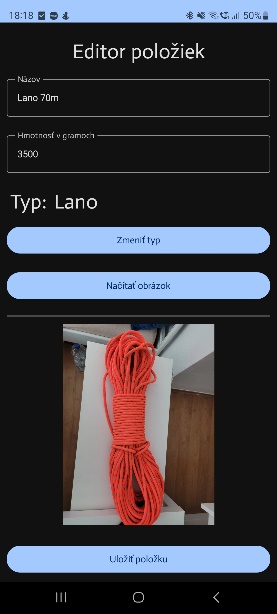
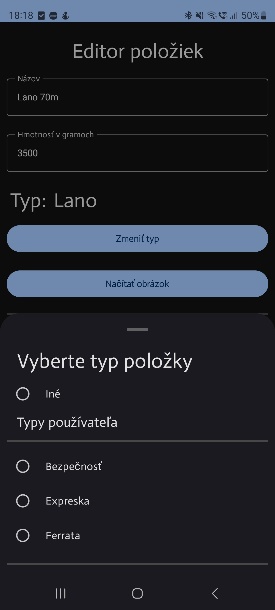
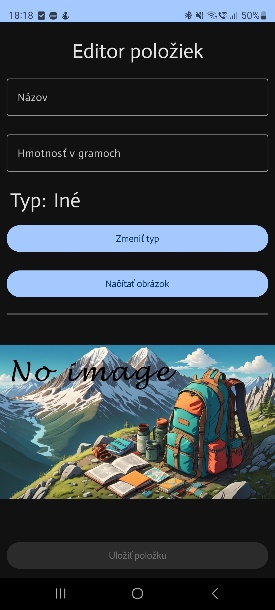
 Obrazovka s predmetmi batohu ponúka podobné možnosti ako obrazovka so všetkými predmetmi.

Tlačidlo vpravo dolu s znakom otázniku slúži ako otvorenie informačného panelu s váhami a % obsadeniami jednotlivých kategórií. Tlačidlo nad ním slúži na vyprázdnenie celého batohu. Ďalšou zmenou je že pri informáciách o predmete je už iba tlačidlo na odstránenie predmetu z batohu. Pribudlo aj tlačidlo vľavo hore zo symbolom srdca. Pomocou tohto tlačidla sa vytvára skupina predmetov. Po jeho stlačení sa zobrazí dialógové okno s textovým poľom pre zadanie mena skupiny. Skupina sa vytvorí z predmetov ktoré sa práve nachádzajú v batohu.

## Obrazovka skupín

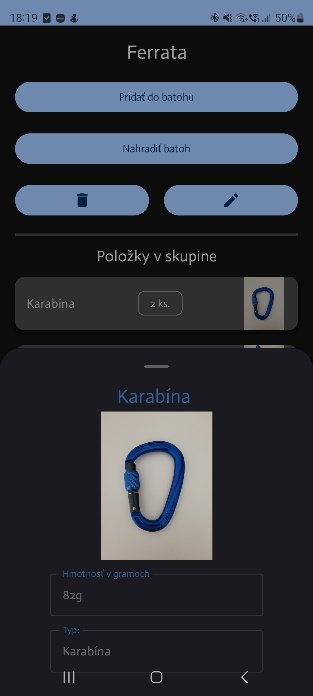
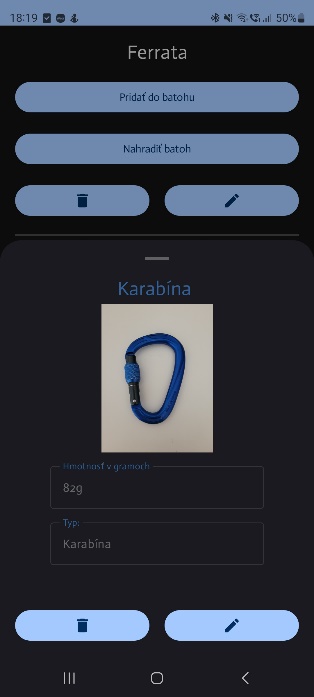
Obrazovka so zoznamom skupín je len jednoduchý zoznam všetkých existujúcich skupín. Vyhľadávanie funguje rovnako ako na ostatných obrazovkách ale neporovnáva sa typ iba meno skupiny. Po stlačení akejkoľvek skupiny sa otvorí informačná obrazovka danej skupiny.

## Editovacia obrazovka

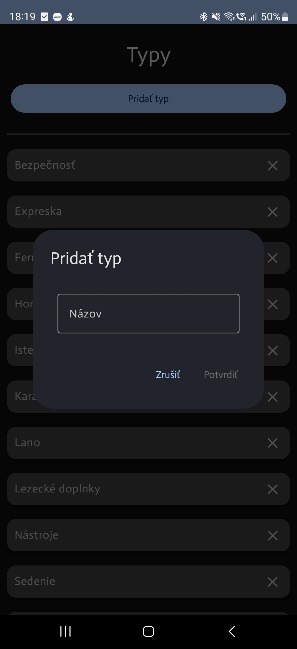
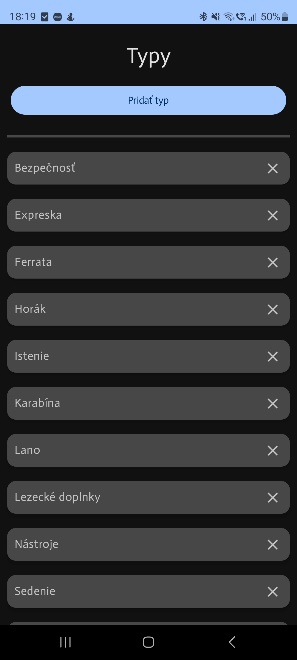


Editovancia obrazovka slúži na vytváranie a upravovanie predmetov. Predmet sa dá uložiť iba v tom prípade že má meno, hmotnosť ktorá je celočíselná a väčšia ako 0 a nahraný obrázok. V prípade že predmet nemá nahraný obrázok alebo nahratý chybný obrázok je na obrazovke kresba s nápisom no image. Používateľ si môže zmeniť typ predmetu tým že stlačí tlačidlo zmeň typ. Po jeho stlačení sa zobrazí malé okno s výberom typov ktoré používateľ vytvoril alebo základným typom Iné (Other). Po stlačení výberu obrázku sa zobrazí vysúvacie okno s výberom obrázku zo zariadenia.

## Informačná obrazovka skupiny

Po kliknutí na skupinu v zozname skupín sa otvorí práve táto obrazovka. Umožňuje používateľovi zistiť aké predmety sa nachádzajú v danej skupine. A upravovať skupinu. Používateľ môže zmeniť názov skupiny tlačidlom s ikonou pera. Zmazať skupinu tlačidlom s ikonou koša. Po kliknutí na ľubovoľný predmet sa zobrazí vysúvacia obrazovka s bližšími informáciami o danom predmete a dvoma tlačidlami na spodku. Prvé s ikonou koša odstráni predmet zo skupiny a druhé s ikonou pera zmení počet daného predmetu v skupine. Počet musí byť celé číslo väčšie ako 0. V skupine sa nachádzajú ešte 2 tlačidlá a to pridanie do batohu ktoré pridá predmety už do existujúcej zostavy predmetov v batohu. Druhé tlačidlo s názvom nahradiť batoh odstráni všetko z batohu a vloží tam predmety zo skupiny. Po stlačení ktoréhokoľvek s týchto dvoch tlačidiel sa aplikácia prepne do obrazovky s predmetmi v batohu.

## Obrazovka typov

Obrazovka typov slúži na jednoduché pridávanie a odstraňovanie typov respektíve kategórii predmetov. Nový typ sa pridáva tlačidlom pridať typ ktoré zobrazí dialógové okno s textovým okno pre názov typu. Pre odstránenie typu treba stlačiť tlačidlo X na pravej strane karty s typom.

# Implementácia

## Room databáza

Room databáza predstavuje jadro aplikácie. Slúži ako uchovávateľ informácii o predmetoch, ich zaradení do batohu. Typoch predmetov a Skupín predmetov. Databáza slúži aj na prácu s danými dátami. Dokáže predmety zo skupiny pridať do batohu. Vymazať všetky záznamy konkrétnej skupiny. Odstrániť predmet aj ako zo skupín tak aj z celkovej databázy. A pod.

@Query("UPDATE items SET addedToBackpack = " +  
 "((SELECT amount FROM groupItems WHERE itemId = items.id and groupName = :groupName) + addedToBackpack)" +  
 "where id in (select itemId from groupItems where groupName = :groupName) ")  
suspend fun addGroupToBackpack(groupName: String)

*Príklad práce zo skupinou pomocou databázy.*

Databáza sa skladá z 3 tabuliek a to tabuľky items, types a groupItems. V celom projekte je využitých 22 rôznych metód nad danou databázou. Tieto metódy sa starajú o update, insert a delete dát.

Databáza je do aplikácie implementovaná pomocou dependency injection. Komunikácia s databázou je vyriešená programovaním voči interface-u. Vďaka čomu je do budúcna možné rozšíriť jednoducho aplikáciu o schopnosť práce s online databázou.

## Navigation

Navigácia v celej aplikácii je riešená pomocou NavHost-u a NavHostControllera. Príkazy pomocou ktorých sa v aplikácii pohybujem medzi jednotlivými obrazovkami sú posúvane obrazovkám ako lambda funkcie. A parametre pre obrazovky sa posúvajú pomocou tried ktoré sú jedináčikmi respektíve typu object. Tieto triedy sú dve jedna sa volá ItemMover a druhá GroupNameMover.

Hlavná aplikácia je vďaka tomu jedna aktivita. A nieje potrebné pre každú obrazovku riešiť inú aktivitu. Jednou ďalšou aktivitou je iba uvítacia obrazovka.

V časti navigácie sa ešte hlavným obrazovkám posúva Composable komponent s názvom bottomBar ktorý obsahuje spodnú navigačnú lištu ktorá sa zobrazuje v hlavných troch obrazovkách.

Použitie NavHost-u umožňuje aj jednoduché a rýchle implementovanie využitie tlačidla späť samotného mobilného zariadenia.

## ViewModel

ViewModel slúži ako uchovávateľ hodnôt a stavov obrazoviek. Každá má jeden. Tak isto sa stará aj o komunikáciu medzi databázou(dáta repozitárom) a používateľským rozhraním. Pracuje aj s už spomenutými triedami ItemMover GroupNameMover.

Nie všetky stavy a hodnoty sa uchovávajú v ňom. ViewModel umožňuje uchovávanie stavov bez zmeny aj medzi jednotlivými prepnutiami obrazovky a lebo otočeniami obrazovky a to bolo v niektorých prípadoch správanie ktoré nebolo požadované. Prvky ktoré bolo požadované neuchovať ich stav naprieč týmito akciami boli preto prenesené priamo do Composable funkcií ako hodnoty uchovávané pomocou remember. To sú napríklad stav otvorenia informačných podobrazoviek a dialógov.

## LifeCycles

LifeCycles bolo použité priamo v mojej práci iba zriedkavo. Ale samotný ViewModel sa dá považovať ako využitie LifeCycles čo vyplýva už zo samotného importu.

import androidx.lifecycle.ViewModel  
import androidx.lifecycle.*viewModelScope*

Ukážkou priameho použitia môže byť prepnutie a delay medzi uvítacou obrazovkou a hlavnou časťou aplikácie.

*lifecycleScope*.*launch* **{** delay(1500)  
 val mainIntent = Intent(this@SplashActivity, MainActivity::class.*java*)  
 startActivity(mainIntent)  
 finish()  
**}**

## Práca s lokálnym úložiskom aplikácie

Aplikácia BackpackManager musí využívať lokálne úložisko aplikácie z dôvodu kopírovania obrázkov predmetov. Tieto obrázky si musí aplikácia kopírovať z dôvodu že ak používateľ daný obrázok vymaže alebo premiestni nebude už uložená cesta k obrázku v databáze platná.

fun Context.copyFileToAppStorage(fileUri: Uri): Uri? {  
 val contentResolver = *contentResolver* val inputStream: InputStream? = contentResolver.openInputStream(fileUri)  
  
 var fileName: String? = null  
 contentResolver.query(fileUri, null, null, null, null)?.*use* **{** cursor **->** val nameIndex = cursor.getColumnIndex(OpenableColumns.*DISPLAY\_NAME*)  
 if (cursor.moveToFirst() && nameIndex != -1) {  
 fileName = cursor.getString(nameIndex)  
 }  
 **}** if (fileName == null) return null  
  
 val outputFile = File(*filesDir*, fileName!!)  
 val outputStream: OutputStream = FileOutputStream(outputFile)  
  
 return try {  
 if (inputStream != null) {  
  
 val buffer = ByteArray(1024)  
 var bytesRead: Int  
 while (inputStream.read(buffer).*also* **{** bytesRead = **it }** != -1) {  
 outputStream.write(buffer, 0, bytesRead)  
 }  
 inputStream.close()  
 }  
 outputStream.close()  
 outputFile.*toUri*()  
 } catch (e: Exception) {  
 null  
 }  
}

*Metóda kopírovania obrázku do súborov aplikácie.*

Z dôvodu že aplikácia si uchováva obrázky v jej súboroch je potrebné aj jednotlivé Composable komponenty ktoré s nimi pracujú prispôsobiť pretože tieto obrázky nepatria do drawable resurces. Pri odstraňovaní predmetov alebo zmene obrázku boli tieto kópie vymazané.

Image(  
 painter = rememberAsyncImagePainter(

model = Uri.parse(item.picturePath),  
 error = painterResource(id = R.drawable.*noimage*),  
 fallback = painterResource(id = R.drawable.*noimage*)  
 ),  
 contentDescription = item.name,  
 modifier = Modifier  
 .*height*(70.*dp*)  
 .*width*(90.*dp*)  
)

*Ukážka Composable komponentu ktorý pracuje s lokálnym obrázkom.*

## Získavanie obrázkov zo zariadenia

Získavanie obrázka respektíve jeho URI je riešené pomocou Intentu ktorý je volaný v tlačidle. Po vybratí obrázku sa uloží jeho URI vo forme stringu do stavu vo ViewModeli.

val galleryLauncher = rememberLauncherForActivityResult(ActivityResultContracts.GetContent()) **{** uri: Uri? **->** editingScreenViewModel.change(editingScreenViewModel

.itemUiState.itemDetails.copy(picturePath = uri.*toString*()))  
**}**

*Launcher ktorý sa stará o získanie URI.*

Button(onClick = **{** galleryLauncher.launch("image/\*") **}**, modifier = Modifier  
 .*fillMaxWidth*()  
 .*padding*(10.*dp*)) **{** Text(text = stringResource(id = R.string.*loadImage*))  
**}**

*Tlačidlo ktoré spúšťa vyššie uvedený launcher.*

# Poznámky

Oproti prvému návrhu boli odstránené nastavenia jazyku a témy z dôvodu že tieto veci si aplikácia berie z nastavení systému, prispôsobuje svoje farby farbám ktoré sú nastavené na telefóne ale v prípade že mobilné zariadenie túto možnosť nepodporuje je vytvorená základná téma. Jazyk bol implementovaný rovnako. Aplikácia má pripravenú anglickú a slovenskú verziu.

# Zdroje

Packing list Travel Packlist: <https://play.google.com/store/apps/details?id=com.tnx.packed&hl=sk&gl=US>

Packingbird – Packing list app: <https://play.google.com/store/apps/details?id=com.packingbird.android&hl=sk&gl=US>

AI pre generáciu obrázkov:  
<https://gencraft.com/generate>

AI pre tvorbu kódu (Použitý pre tvorbu ukážkového kódu ako použiť a a pracovať s rôznymi prvkami, kód priamo z neho nebol nikde použitý):  
<https://chatgpt.com/>

Tvorba základnej témy:  
<https://material-foundation.github.io/material-theme-builder/>

Ukážkové kódy a použitia pre Jetpack Compose komponenty:  
<https://m3.material.io/components>