ŽILINSKÁ UNIVERZITA V ŽILINE
Fakulta riadenia
a informatiky

Semestrálna práca z predmetu

# VÝVOJ APLIKÁCIÍ PRE MOBILNÉ ZARIADENIA

**vypracoval**: Quang Phung Viet

**študijná skupina**: 5ZYR21

**cvičiaci**: doc. Ing. Patrik Hrkút, PhD.

**termín cvičenia**: streda Blok 1-3

v Žiline dňa 01.05.2024

## Špecifikácia zadania:

Za semestrálnu prácu som si vybral vypracovať aplikáciu na nákupný zoznam. Výber tejto témy bol z dôvodu malého výskytu na app store Obchod Play kde som videl potenciálnu medzeru, ktorú by som mohol doplniť. Zameral som sa viac na finančnú stránku nakupovania. Teda prehľad návykov nakupovania. Za koľko bol zaplatený nákup, a aj krivka priemeru nákupov. A pre doplnok som pridal aj hlasový vstup pre niektoré polia, ktoré musí užívateľ vyplniť.

## Podobné aplikácie a popis:

Nezmenilo sa moc od poslaného checkpointu, takže tu skopírujem požadované veci na dokumentáciu.

Listonic:

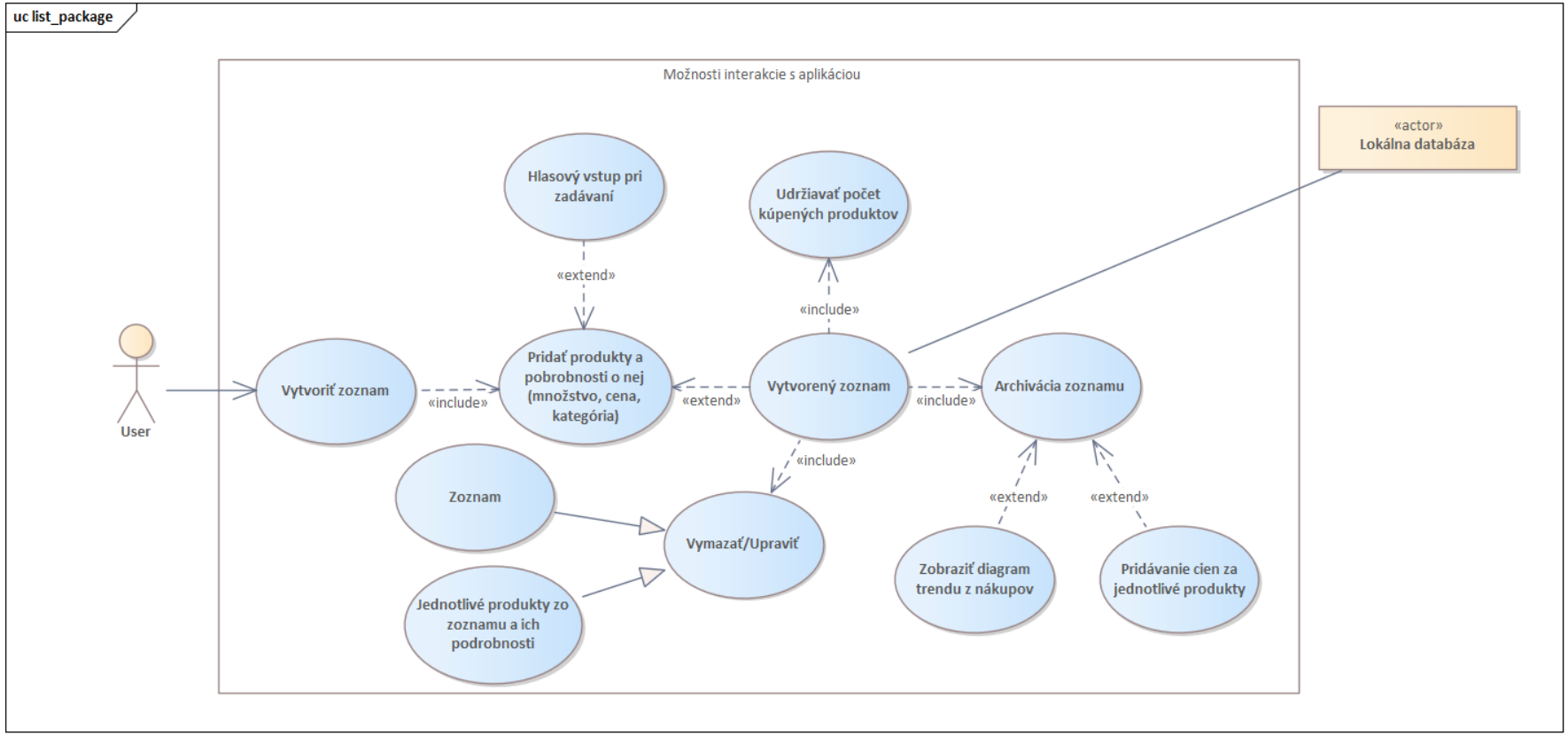
- Táto aplikácia má najbližšie k mojej verzií shopping listu.
- Vyznačuje sa možnosťou prístupu cez webovú stránku listonic.com a zdieľanie zoznamov s ostatnými (musia mať tiež aplikáciu).
- Hlavným publikom sú ľudia, ktorí chcú mať zorganizovaný list na nákup.
- Moja aplikácia je viac orientovaná na financie, čiže bude ukazovať cenové trendy v nákupoch (diagram).
- Mínusy mojej apky oproti tejto bude hlavne prístup k zoznamu cez web a zdieľanie zoznamu s ostatnými.

Bring!

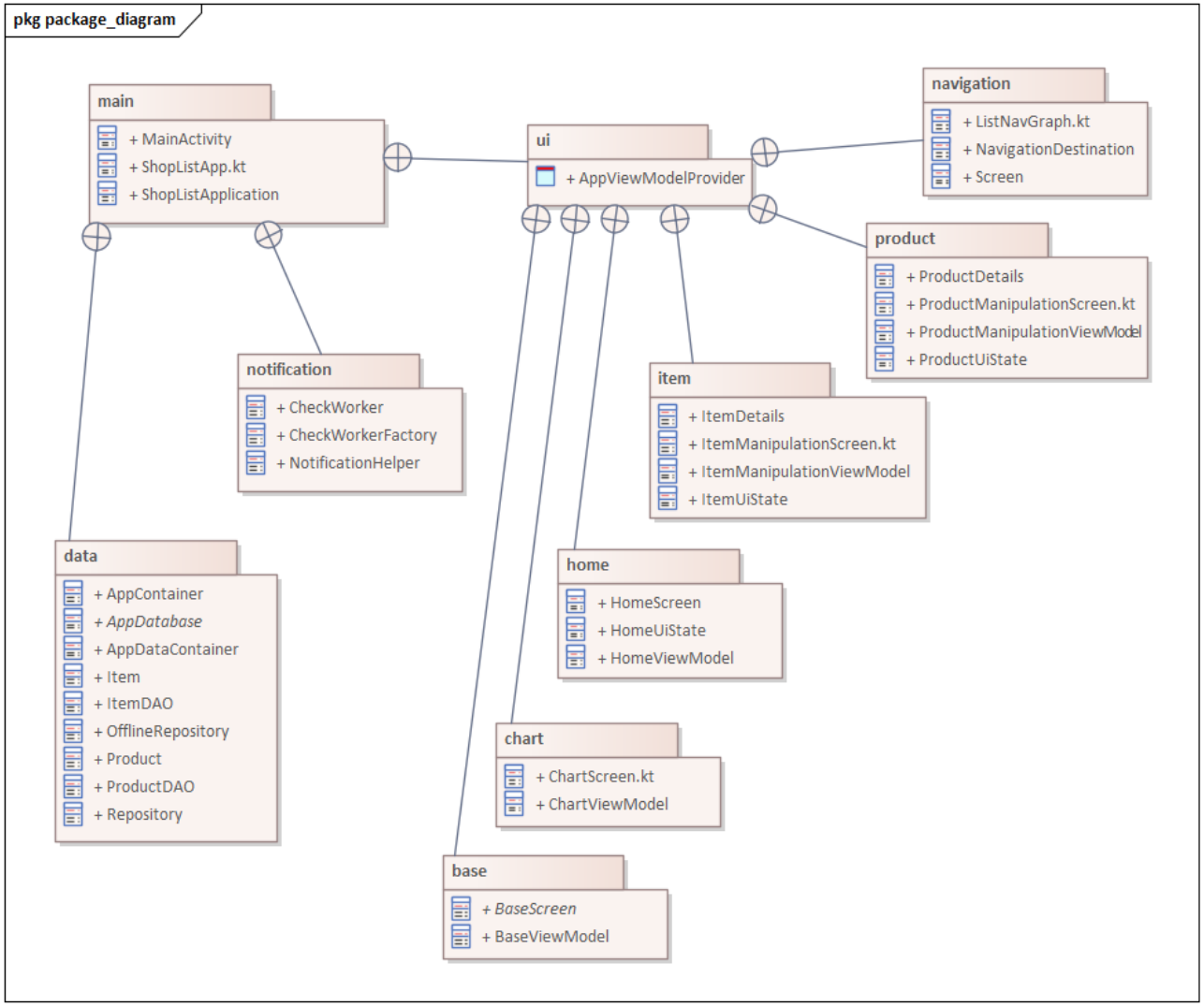
- Viac menej je obsah aplikácie rovnaký ako Listonic, ale líši sa funkciami ako:
- Možnosť pridávať zákaznícke karty
- Pridávať recepty s potrebnými ingredienciami

## Návrh riešenia problému:

Krátka analýza:



Návrh aplikácie:




Podrobnejšie ku jednotlivým triedam bude k dispozícií v repozitári súbor .eap.

## Popis implementácie:

Obrazovky:

Tri obrazovky. „Hlavná obrazovka“ je samotný nákupný zoznam. Následne ďalšia obrazovka je „archivačná“, kde sa ukladajú už splnené/dokončené zoznamy. A posledná obrazovka ukazuje graf trendu nákupov.

Využitie AndroidX komponentov:

Room (celý priečinok „data“ je implementácia databázy),

ViewModel (jednotlivé ViewModely popri „screen“ triedach),

Navigation(priečinok navigation)

Paging

```
override fun getVisibleItemsStream(): Flow<PagingData<Item>> {
    return Pager(PagingConfig(pageSize = 50)) {
        itemDao.getVisibleItems()
    }.flow
}

override fun getNonVisibleItemsStream(): Flow<PagingData<Item>> {
    return Pager(PagingConfig(pageSize = 50)) {
        itemDao.getNonVisibleItems()
    }.flow
}
```

LifeCycles

```
LaunchedEffect(key1 = chartViewModel) {
    chartViewModel.fetchInvisibleItems()
}
```

WorkManager

```
val workRequest = PeriodicWorkRequestBuilder<CheckWorker>(1,
TimeUnit.DAYS).build()
Configuration.Builder()
    .setWorkerFactory(CheckWorkerFactory(container))
    .build()
WorkManager.getInstance(this).enqueue(workRequest)
```

Notifikácia:

Upozorniť používateľa po jednom dni ak nezadal do archivovaného itemu ceny za jednotlivé produkty.

Po týždni pripomenúť používateľa (ak sú dáta pre diagram dostupné), aby si pozrel trend svojich nákupov.

```
if (numberOfItemsWithTotal <= numberOfArchivedItems && currentDate !=
lastDailyReminderDate) {
    val notificationHelper = NotificationHelper(applicationContext)
    val subtractionPom = numberOfArchivedItems - numberOfItemsWithTotal
    notificationHelper.createNotification("Reminder", "Please enter prices
for your products in $subtractionPom items ! ", 1)
    sharedPreferences.edit().putLong("lastDailyReminderDate",
currentDate).apply()
    return Result.success()
}
```

```
if (numberOfItemsWithTotal > 5 && currentDate - lastWeeklyReminderDate >=
TimeUnit.DAYS.toMillis(7)) {
    val notificationHelper = NotificationHelper(applicationContext)
    notificationHelper.createNotification("Reminder", "Check your shopping
trends!", 2)
    sharedPreferences.edit().putLong("lastWeeklyReminderDate",
currentDate).apply()
    return Result.success()
}
```

Použitie externého frameworku / knižnice:

Voice Input

```
private val _voiceInput = MutableStateFlow("")
val voiceInput = _voiceInput.asStateFlow()

fun handleVoiceInputResult(data: Intent?) {
    val matches =
data?.getStringArrayListExtra(RecognizerIntent.EXTRA_RESULTS)
    if (!matches.isNullOrEmpty()) {
        val voiceInput = matches[0]
        viewModelScope.launch {
            _voiceInput.emit(voiceInput)
        }
    }
}
```

Chart API (https://github.com/patrykandpatrick/vico)

```
Chart(
    chart = composedChart,
    chartModelProducer = composedChartEntryModelProducer,
    modifier = modifier,
    startAxis = rememberStartAxis(valueFormatter =
integerAxisValueFormatter,
        title = "Product No."),
    bottomAxis = rememberBottomAxis(),
    endAxis = rememberEndAxis(title = "y = Price $", titleComponent =
textComponent { this.textSizeSp = 16f },
        label = null, tick = null, itemPlacer =
AxisItemPlacer.Vertical.default(maxItemCount = 0)),
    topAxis = rememberTopAxis(title = "x = item No.", titleComponent =
textComponent { this.textSizeSp = 16f },
        label = null, tick = null)
)
```

## Zoznam použitých zdrojov:

https://github.com/patrykandpatrick/vico/tree/master/sample/src/main/java/com/patrykandpatrick/vico/sample/showcase/charts