ŽILINSKÁ UNIVERZITA V ŽILINE

FAKULTA RIADENIA A INFORMATIKY

Semestrálna práca

Vypracoval: Alan Daniel Šupina

Predmet: Vývoj aplikácií pre mobilné zariadenia

Cvičiaci: Ing. Michal Ďuračík, PhD. a doc. Ing. Patrik Hrkút, PhD.

Študijná skupina: 5ZYI23

Akademický rok: 2023

Ročník: Druhý

1. Popis a analýza riešeného problému

1.1. Špecifikácia zadania

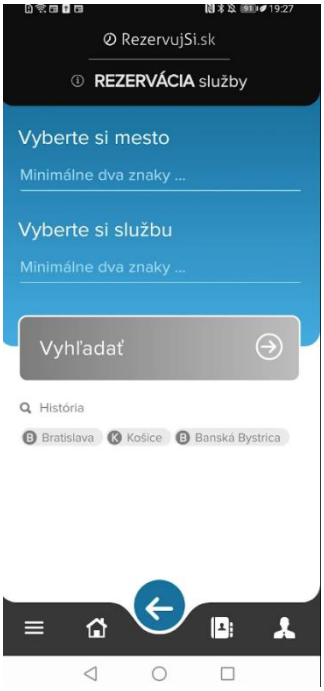
Moja aplikácia slúži na rezervovanie miest na sedenie v rôznych podnikoch. Keď užívateľ zapne aplikáciu, tak v domovskej stránke dostane na výber dve možnosti. Buď si vie prezrieť svoje rezervácie alebo môže vykonať novú rezerváciu.

Ak si vyberie možnosť prezretia svojich rezervácii, tak ho to presunie na ďalšiu obrazovku (aktivitu), kde bude mať usporiadané svoje rezervácie. Pre každú rezerváciu bude vypísané: názov podniku, názov mesta, v ktorom sa podnik nachádza a deň, na kedy je naplánovaná rezervácia.

Pokiaľ si vyberie uskutočniť ďalšiu rezerváciu, tak ho to presunie na ďalšiu obrazovku, kde podniky sú vypísané pod sebou a pre každý podnik je vypísaný jeho názov a mesto, v ktorom sa nachádza.

Keď si užívateľ vyberie konkrétny podnik, tak ho to presunie na ďalšiu obrazovku, v ktorej okrem názvu daného podniku a názvu mesta, bude mať aj formulár, v ktorom bude zadávať deň rezervácie.

1.2. Spracovanie prehľadu dostupných aplikácií podobného zameranie



Aplikácia RezervujSi.sk klient je podobná ako tá moja. Tiež dokáže uskutočňovať rezervácie v daných podnikoch na daný čas. Táto aplikácia dokáže navyše v daných podnikom si rezervovať konkrétnu službu, takže sa zameriava na firmy ako pneuservisy, masáže a tak ďalej. Moja aplikácia má v budúcnosti sa zameriavať na rezerváciu miest na sedenie na konkrétny čas v podnikoch, reštauráciách a nočných kluboch. Taktiež do mojej aplikácie bude pridaná možnosť objednávky jedla na presný čas, ak to bude reštaurácia. Tým pádom, keďže aplikácia RezervujSi.sk klient ponúka aj výber konkrétnej služby, tak vo vyhľadávaní sa dokáže filtrovať nie len pomocou mesta, ale aj pomocou služby.

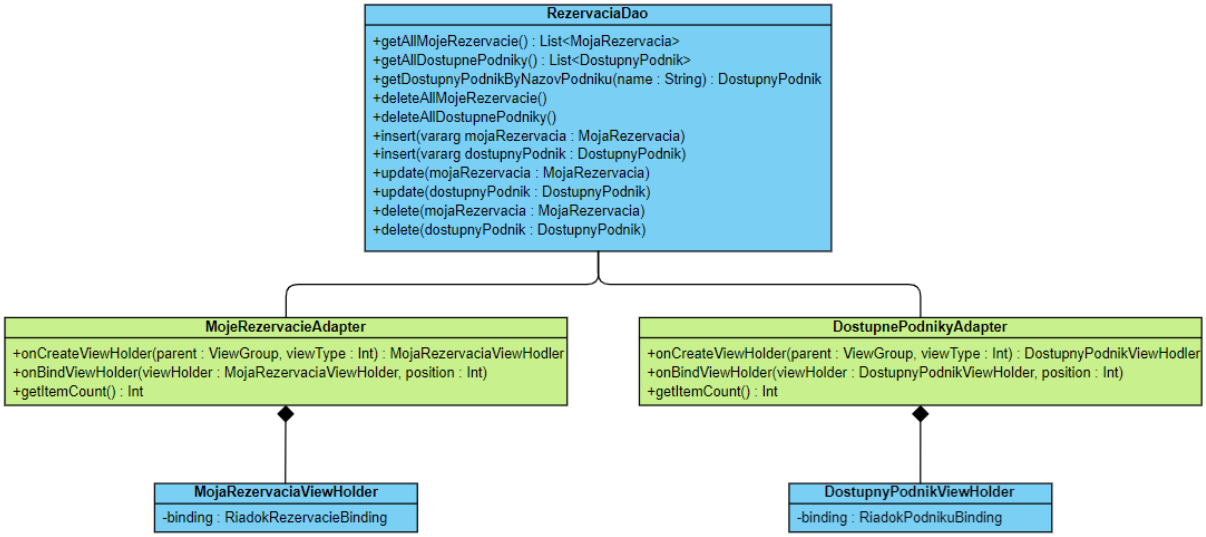
Aplikácia Wolt je veľmi známa a umožňuje objednávku jedla z reštaurácií na danú adresu. Hlavný rozdiel v tejto aplikácii oproti tej mojej je ten, že Wolt síce vypisuje, že za koľko by mohla byť objednávky doručená, ale nedá sa objednať jedlo na presný čas. Moja aplikácia dokáže rezervovať na presný čas v danom podniku. Spoločné je tu to, že Wolt tiež dokáže ukázať

2. Návrh riešenia problému

2.1. Prípady použitia

- zobrazenie všetkých rezervácií
- prehľadávanie dostupných podnikov v určitých mestách
- uskutočnenie novej rezervácie v danom podniku na určitý deň

2.2. UML diagram tried



3. Popis implementácie

3.1. Aktivity

V aplikácie som použil celkovo 4 aktivity. Jedna aktivita slúži ako vstupný bod aplikácie, kde má užívateľ na výber, či si chce pozrieť svoje rezervácie alebo chce uskutočniť novú rezerváciu. Ďalšie dve aktivity slúžia ako zoznamy s využitým adaptérov pre zobrazenie jednotlivých rezervácií a podnikov. Posledná štvrtá aktivita slúži na vytvorenie novej rezervácie na určitý deň v danom podniku.

3.2. Uchovanie stavu aktivity pred otočením

Na uchovanie stavu aktivity pred otočením využívam Bundle, ktorý v sebe uchováva kľúč a hodnotu. V aktivite VybranyPodnikActivity používam túto technológiu na uloženie vstupu od užívateľa v podobe dňa rezervácie.

```
override fun onSaveInstanceState(outState: Bundle) {
    super.onSaveInstanceState(outState)

    val binding = ActivityVybranyPodnikBinding.inflate(layoutInflater)
    outState.putString("nazovDna", binding.inputDen.text.toString())
}
```

Obrázok 1 Metóda na uloženie stavu aktivity

```
if (savedInstanceState != null) {
    val nazovDna = savedInstanceState.getString( key: "nazovDna",  defaultValue: "den")
    binding.inputDen.text = Editable.Factory.getInstance().newEditable(nazovDna)
}
```

Obrázok 2 Blok príkazov pre načítanie stavu aktivity

3.3. Data binding

V aplikácii taktiež využívam Data binding pre asociáciu views.

```
val binding = ActivityVybranyPodnikBinding.inflate(layoutInflater)
setContentView(binding.root)
```

Obrázok 3 Data binding

3.4. Room

V aplikácie pre ukladanie dát využívam knižnicu Room, v ktorej uchovávam dve tabuľku a to tabuľku pre rezervácie a tabuľku pre podniky.

```
@Database(entities = [DostupnyPodnik::class, MojaRezervacia::class], version = 1)
abstract class RezervaciaDatabase : RoomDatabase() {
    alandanielsupina
    abstract fun rezervaciaDao(): RezervaciaDao

    alandanielsupina
    companion object {
        @Volatile
        private var INSTANCE: RezervaciaDatabase? = null

        /**
         * Metóta vráti inštanciu tiredy
         *
         * @param context
         */
        alandanielsupina
        fun getInstance(context: Context): RezervaciaDatabase {
            return INSTANCE ?: synchronized( lock: this) {
                INSTANCE ?: Room.databaseBuilder(
                    context.applicationContext,
                    RezervaciaDatabase::class.java,  name: "rezervacia_database"
                ) Builder<RezervaciaDatabase!>
                    .fallbackToDestructiveMigration()
                    .allowMainThreadQueries().build() RezervaciaDatabase
                    .also { INSTANCE = it }
            }

        }
    }
}
```

Obrázok 4 Súbor RezervaciaDatabase

```
alandanielsupina
@Dao
interface RezervaciaDao {

    /**
     * Metóda vráti všetky rezervácie
     *
     * @return  rezervacie
     */
    alandanielsupina
    @Query("SELECT * FROM moje_rezervacie")
    fun getAllMojeRezervacie(): List<MojaRezervacia>
    /**
     * Metóda vráti všetky podniky
     *
     * @return  podniky
     */
    alandanielsupina
    @Query("SELECT * FROM dostupne_podniky")
    fun getAllDostupnePodniky(): List<DostupnyPodnik>

    /**
     * Metóda vráti podnik podľa názvu podniku
```

Obrázok 5 Súbor RezervacieDao

```
@Entity(tableName = "moje_rezervacie")
data class MojaRezervacia (
    @PrimaryKey(autoGenerate = true) val id: Int,
    val nazov_podniku : String,
    val nazov_mesta : String,
    val den : String
    )
```

Obrázok 6 Súbor MojaRezervacia

```
@Entity(tableName = "dostupne_podniky")
data class DostupnyPodnik(
    @PrimaryKey(autoGenerate = true) val id: Int,
    val nazov_podniku : String,
    val nazov_mesta : String
)
```

Obrázok 7 Súbor DostupnyPodnik

3.5. RecyclerView

V aplikácii som dvakrát využil zoznam s adaptérom, pričom jeden zoznam slúži pre zobrazenie rezervácií a ten druhý na zobrazenie dostupných podnikov.

V zozname dostupných podnikov využívam funkcionalitu kliknutia na určitý podnik a následne presmerovanie na uskutočnenie rezervácie. Na tento problém som využil Lamdu funkciu.

```
binding.dostupnePodnikyZoznam.adapter = DostupnePodnikyAdapter(zoznamPodnikov) { dostupnyPodnik ->
    val prefs = PreferenceManager.getDefaultSharedPreferences( context: this)
    prefs.edit().putString("nazovVybranehoPodniku", dostupnyPodnik.nazov_podniku).apply()
    startActivity(Intent( packageContext: this, VybranyPodnikActivity::class.java))
}
```

Obrázok 8 Ukážka vytvorenia adaptéra s Lamda funkciou

3.6. Ošetrenie nesprávnych vstupných dát

Na ošetrenie nesprávnych vstupných dát využívam technológiu Toast, pomocou ktorej dávam užívateľovi vedieť, že zadáva nesprávne údaje.

```
binding.potvrditBtn.setOnClickListener { it: View!
    if (binding.inputDen.text.toString().trim().isNotEmpty()) {
        when (binding.inputDen.text.toString()) {
            "Pondelok" -> pridajRezervaciu( den: "Pondelok", vybranyPodnik, rezervaciaDao)
            "Utorok" -> pridajRezervaciu( den: "Utorok", vybranyPodnik, rezervaciaDao)
            "Streda" -> pridajRezervaciu( den: "Streda", vybranyPodnik, rezervaciaDao)
            "Stvrtok" -> pridajRezervaciu( den: "Štvrtok", vybranyPodnik, rezervaciaDao)
            "Piatok" -> pridajRezervaciu( den: "Piatok", vybranyPodnik, rezervaciaDao)
            "Sobota" -> pridajRezervaciu( den: "Sobota", vybranyPodnik, rezervaciaDao)
            "Nedela" -> pridajRezervaciu( den: "Nedeľa", vybranyPodnik, rezervaciaDao)
            else -> Toast.makeText( context: this,  text: "Nesprávny formát dňa", Toast.LENGTH_SHORT).show()
        }
    } else {
        Toast.makeText( context: this,  text: "Zadajte deň", Toast.LENGTH_SHORT).show()
    }
}
```

Obrázok 9 Ukážka ošetrenie chybných vstupov

4. Zoznam použitých zdrojov

- https://developer.android.com/