



ŽILINSKÁ UNIVERZITA V ŽILINE
Fakulta riadenia
a informatiky

SEMESTRÁLNA PRÁCA

HiveMind



Vypracoval: Oliver Pavlanin

Študijná skupina: 5ZYR22

Predmet: Vývoj aplikácií pre Mobilné zariadenia

Obsah

1	Popis a analýza riešeného problému	3
1.1	Špecifikácia zadania	3
1.1.1	Účel aplikácie	3
1.1.2	Cieľová skupina.....	3
1.2	Definovanie problému	3
1.3	Porovnanie s aplikáciami podobného zamerania.....	4
1.3.1	Bee hive monitoring	4
1.4	ApiManager	4
1.5	HiveTracks.....	5
2	Návrh riešenia problému	5
2.1	Analýza aplikácie	5
2.2	Návrh Aplikácie.....	7
2.3	Implementácia	8
2.3.1	Aktivity a fragmenty	8
2.3.1.1	HomeFragment	8
2.3.1.2	HiveFragment	8
2.3.1.3	InHive Fragment	9
2.3.1.4	Cure Fragment	9
2.3.1.5	Inspection Fragment.....	10
2.3.1.6	Monitoring Fragment	10
2.3.1.7	Graph Fragment	11
2.3.2	Broadcast receiver	11
2.3.3	DataBinding	12
2.3.4	Navigation	12
2.3.5	AlarmManager	13
2.3.6	Widget	14
2.3.7	Notifikácie	14
2.3.8	Dynamické rozloženie	14
2.3.9	GPS.....	14
2.3.10	Bluetooth	15
2.3.11	Štýly	15

1 Popis a analýza riešeného problému

Táto kapitola semestrálnej práce slúži na priblíženie aplikácie BeeMind, jej určenie, účel a rozobratie problému.

1.1 Špecifikácia zadania

V tejto časti dokumentácie je rozobraný účel aplikácie, cieľová skupina, pre ktorú je táto aplikácia určená a prípady jej použitia.

1.1.1 Účel aplikácie

Táto aplikácia bude slúžiť ako užívateľské rozhranie pre monitorovaciu jednotku umiestnenú v „srdci“ úľa. Monitorovanie bude schopné sledovať aktuálnu váhu, teplotu a vlhkosť v úli. V aplikácii sa bude taktiež nachádzať história zásahov do úľa, liečenie úľa a monitorovanie. Aplikácia bude taktiež schopná vytvárať štatistiky z jej dostupných údajov.

1.1.2 Cieľová skupina

Táto aplikácia by mohla byť určená pre včelárov a včelárske organizácie, ktoré potrebujú zaznamenávať údaje o zásahoch do úľa.

Začínajúcim včelárom, pre poskytovanie stavu svojich úľov a na získanie dôležitých informácií o živote včiel.

1.2 Definovanie problému

Problém, ktorý táto aplikácia rieši je náhrada ručného zapisovania údajov na papier, keďže dokáže prehľadnejšie zapisovať ako aj zobrazovať históriu zásahov a liečení včiel v jednotlivých úľov.

Taktiež aplikácia poskytuje včelárovi aj informácie zo „srdce“ úľa, vďaka ktorým je schopný sledovať množstvo medu nachádzajúce sa v úli poprípade sleduje možnosť vyrojenia sa včiel.

1.3 Porovnanie s aplikáciami podobného zamerania

V tejto časti sa rozoberajú výhody aplikácií podobného zamerania s aplikáciou BeeMind.

1.3.1 Bee hive monitoring



Obrázok 1 bee hive monitoring app logo

Aplikácia načíta nový včelí úl pomocou QR kódu. Následne sa zobrazí zoznam už pridaných úl'ov, kde pre každý je zobrazený jeho stav, prítomnosť včiel, prítomnosť matky a rojenie. Taktiež je možné zobraziť jednotlivé štatistiky zobrazené v grafoch napr. hmotnosť úl'a alebo jeho teplota. Taktiež je zobrazený aj stav batérie monitorovacieho zariadenia sledovaného úl'a.

Oproti našej aplikácii, táto slúži výhradne ako GUI, pre aktuálne zobrazenie aktuálnych informácií o včel'om úli no nemá možnosť zapisovať zásahy a liečenia včiel.

+

Pridávanie úl'ov pomocou QR kódu

Sledovanie prítomnosti matky

Stav batérie

-

Zapisovanie zásahov

Zapisovanie liečení

Pripomienky

1.4 ApiManager

Aplikácia, ktorá taktiež slúži na sledovanie včelích úl'ov a zaznamenávanie informácií o nich. Slúži skôr ako zapisovač a plánovač s rozvrhom. Záznamy sú zadávané užívateľom, čiže aplikácia nekomunikuje s monitorom nachádzajúcim sa v úli.

Oproti našej aplikácii disponuje podrobnejšími informáciami o úl'och, no nemá možnosť komunikácie s riadiacou jednotkou nachádzajúcej sa v úli.



Obrázok 2 ApiManager app logo

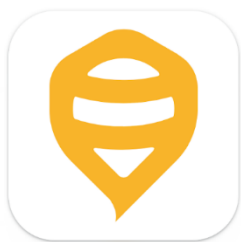
+

Podrobné informácie o úli

-

monitoring úl'a
grafy a štatistiky

1.5 HiveTracks



Obrázok 3 HiveTracks app logo

Táto aplikácia taktiež slúži len na plánovanie úloh a vytváranie štatistik a pripomienok pre jej používateľa. Aplikácia užívateľovi umožňuje efektívne sledovať zdravie včiel v úľoch a plánovač s históriou.

Oproti našej aplikácii disponuje mierne podrobnejšími manuálne zadávanými informáciami o úli. Na druhej strane taktiež slúži len ako zapisovač a plánovač bez možnosti automatického sledovania informácií o úli.

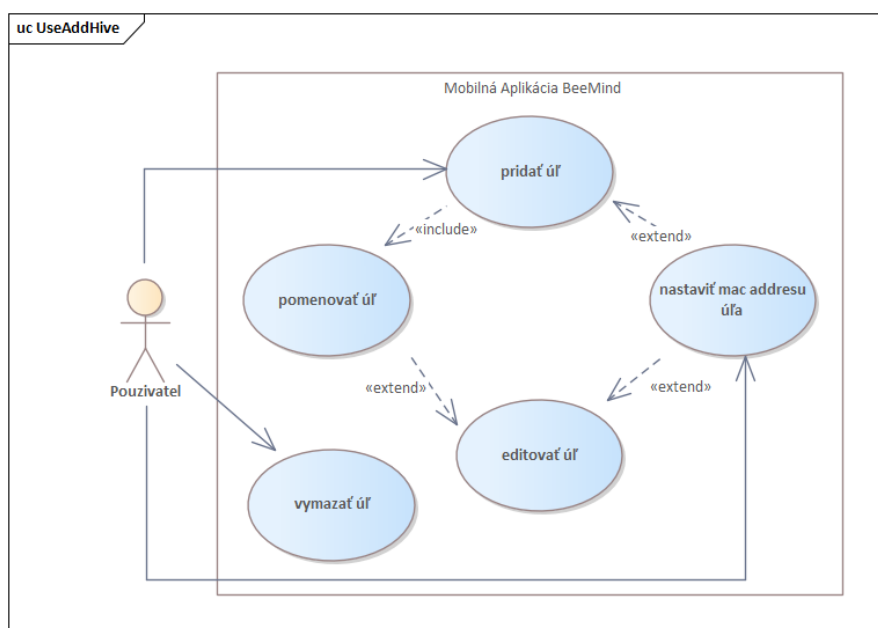
+	-
dizajn	monitoring úľa
rady	grafy a štatistiky

2 Návrh riešenia problému

V tejto časti dokumentácie sú pomocou diagramov prípadov použitia vysvetlené možnosti aplikácie, ktoré sú podložené snímkami obrazovky z aplikácie.

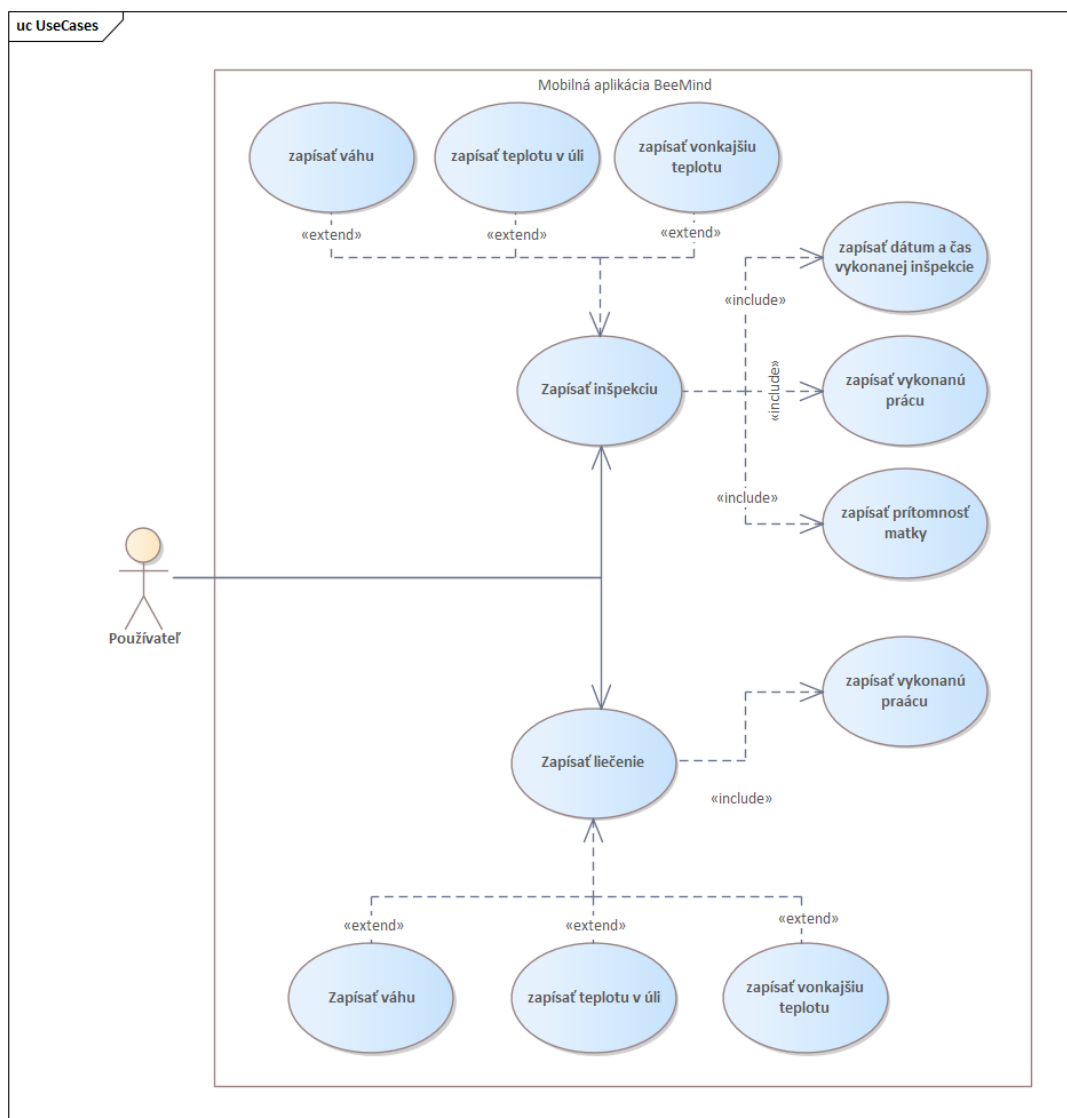
2.1 Analýza aplikácie

Používateľ môže spravovať včelín, čiže pridávať, editovať a vymazávať úle vo včelíne. Pri pridávaní úľa musí používateľ zadať jeho názov, v prípade, že úľ obsahuje riadiacu jednotku zadá aj jej MAC adresu. Pri editovaní používateľ môže zmeniť názov úľa poprípade zmeniť resp. pridať MAC adresu riadiacej jednotky. (Vid' obrázky 8,9,10)



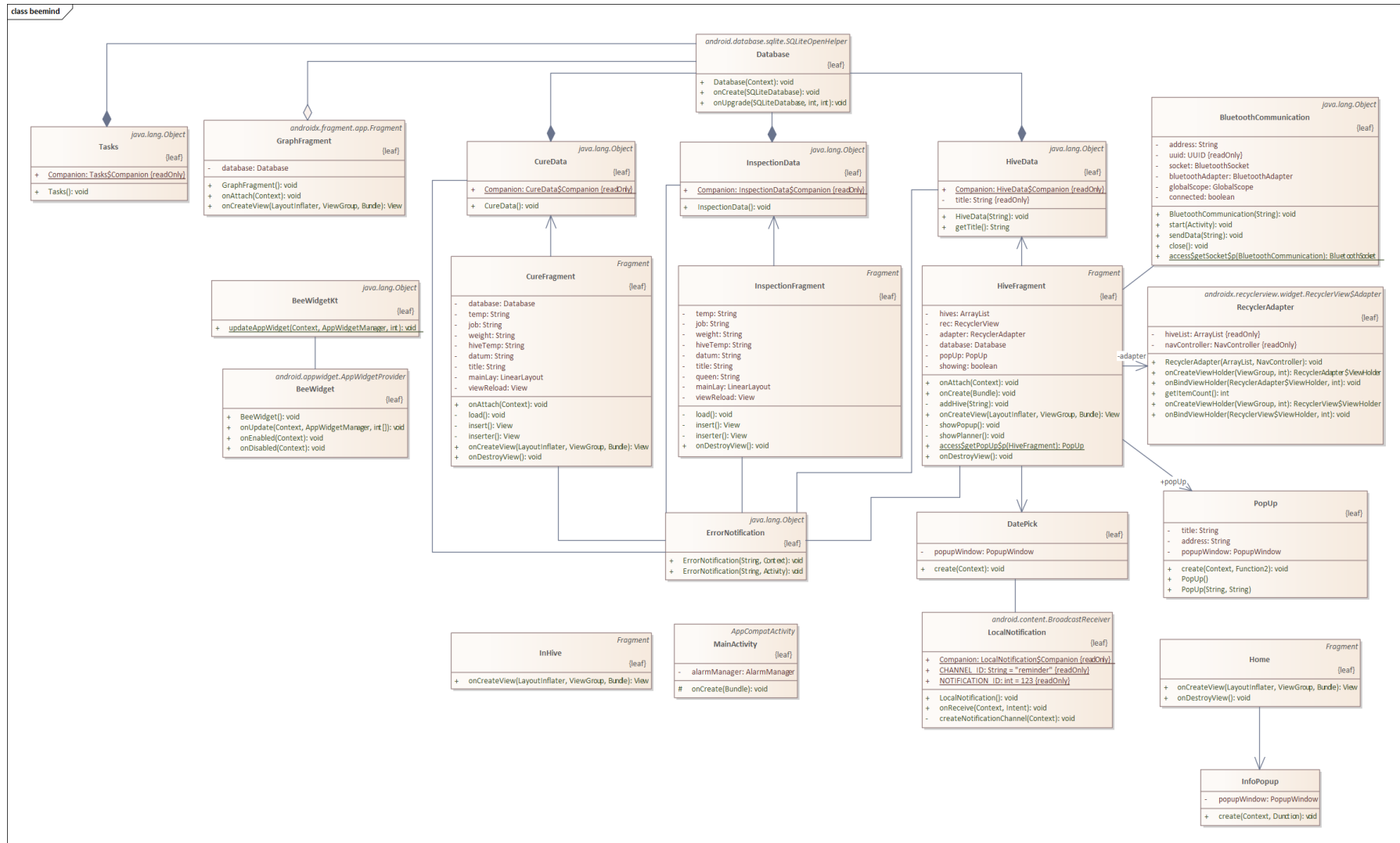
Obrázok 4 Diagram prípadov použitia pre spravovanie úľov

Taktiež používateľ môže zadávať záznamy o liečení aj o inšpekcií. Pri liečení sú dva povinné záznamy, ktoré používateľ musí zadať, a to dátum, čas(V prípade nevyplnenia sa automatický doplní aktuálny čas) a popis vykonanej práce.



Obrázok 5 Diagram prípadov použitia pre pridávanie záznamov

2.2 Návrh Aplikácie



Obrázok 6 Diagram tried

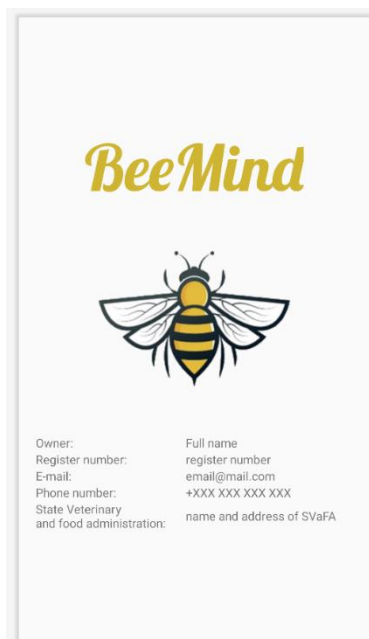
2.3 Implementácia

V tejto časti sa popisujú využité komponenty a technológie, a zobrazenie ich implementácie v tejto semestrálnej práci.

2.3.1 Aktivity a fragmenty

V aplikácii sa nachádza 7 fragmentov a to:

2.3.1.1 HomeFragment



Obrázok 7 Home Fragment

Predstavuje úvodnú stránku aplikácie. Fragment tvorí nadpis aplikácie, tlačidlo s obrázkom loga a Informácie o majiteľovi včelnice. Vo fragmente sa nachádza:

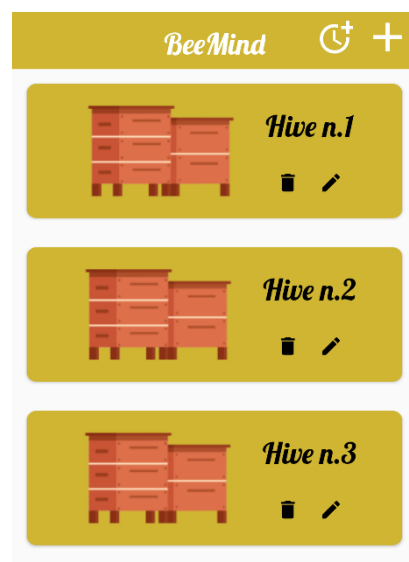
1. TextView s názvom aplikácie - tento prvok zobrazuje názov aplikácie a je umiestnený v hornej časti fragmentu.
2. Tlačidlo s obrázkom loga. Po kliknutí na toto tlačidlo prejdeme do HiveFragmentu.
3. Tabuľka s informáciami o majiteľovi - tento prvok zobrazuje informácie o majiteľovi včelína ako Meno a priezvisko vlastníka, registračné číslo, email, telefónne číslo a štátnu veterinárnu a potravinovú správu.

2.3.1.2 HiveFragment

HiveFragment slúži na spravovanie úľov a úloh. V tomto fragmente je možné pridať úľ, editovať jeho názov alebo MAC adresu a vymazať daný úľ.

Taktiež sa tu nachádza možnosť pridať pripomienku vo forme notifikácie a to buď vo zvolený čas alebo pri príchode na včelnicu. Vo fragmente sa nachádza:

1. Toolbar s názvom aplikácie a dvoma tlačidlami pre pridanie úľa a pre pridanie pripomienky. Po ich stlačení sa zobrazí vyskakovanie okno s nastaveniami daných parametrov.
2. RecyclerView, v ktorom sa nachádzajú jednotlivé úle uložené v CardView. Po kliknutí na jednotlivý úľ prejdeme do InHive Fragmentu



Obrázok 8 Hive Fragment

Enter hive name

hive title

mac address

Obrázok 9 Vyskakovanie okno pre pridávanie nového úľa

Create new task

Apr 12 2022

May 13 2023

Jun 14 2024

07 00

08 01

09 02

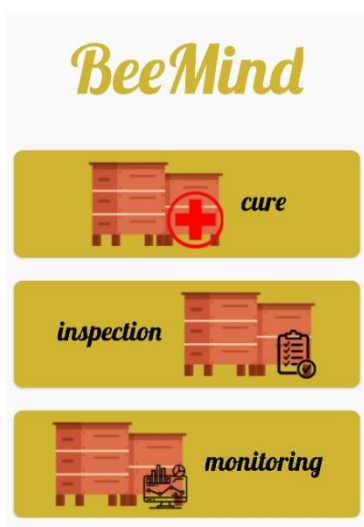
task description

CONFIRM

☐ on arrive

Obrázok 10 Vyskakovanie okno pre pridávanie novej úlohy

2.3.1.3 InHive Fragment



Obrázok 11 InHive Fragment

Tento fragment slúži len na prepínanie sa medzi tromi ďalšími Fragmentami. Nachádzajú sa v ňom:

1. Názov aplikácie
2. 3 „Karty“ a to liečenie, inšpekcia a monitorovanie, kde po kliknutí na každú z nich prejdeme do príslušného fragmentu (Z hora cureFragment, InspectionFragment a MonitoringFragment).

2.3.1.4 Cure Fragment

Slúži na pridávanie záznamov o liečení včelstvá v jednotlivých úľoch. Vo fragmente sa nachádzajú:

1. TableView s názvami jednotlivých stĺpcov.
2. TableView s históriou záznamov.
3. TableView s EditTextami pre vkladanie záznamov.
4. Tlačidlo, ktoré preformátuje vložené údaje a vloží ich do databázy.

Date	Temp	Job Description	Weight	HiveTemp
13.05.23 08:57	11.3	Pridaná 1xS	33.5	24.6
dd.mm.yy HH:MM	Temp	Job description	Weight	HiveTemp
+				

Obrázok 12 Cure Fragment

2.3.1.5 Inspection Fragment


Rovnako ako v Cure Fragmente sa tu nachádzajú záznamy o vykonaných inšpekciách.

Vo fragmente sa nachádzajú:

1. TableView s názvami jednotlivých stĺpcov.
2. TableView s históriou záznamov.
3. TableView s EditTextami pre vkladanie záznamov.
4. Tlačidlo, ktoré preformátuje vložené údaje a vloží ich do databázy.

Vo fragmente sa nachádzajú:

1. TableView s názvami jednotlivých stĺpcov.
2. TableView s históriou záznamov.
3. TableView s EditTextami pre vkladanie záznamov.
4. Tlačidlo, ktoré preformátuje vložené údaje a vloží ich do databázy.

<i>Date</i>	<i>Temp</i>	<i>Job Description</i>	<i>Weight</i>	<i>HiveTemp</i>	<i>Q</i>
13.05.23 09:13	13.1	Pridaná 1xS	38	24.2	M+
dd.mm.yy HH:MM	Temp	Job description	Weight	HiveTemp	Q
					

Obrázok 13 Inspection Fragment

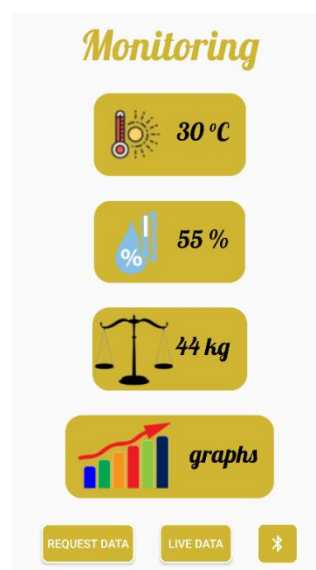
2.3.1.6 Monitoring Fragment

Monitoring Fragment slúži na zobrazenie aktuálnych dát z úľa a na komunikáciu s úľom cez Bluetooth.

Obsahuje 3 tlačidlá pre:

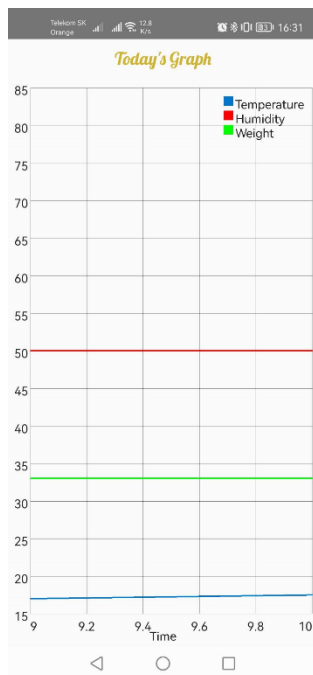
1. Pripojenie sa na riadiacu jednotku úľa.
2. Požiadanie o aktuálne dáta.
3. Požiadanie o dáta od poslednej žiadosti.

Ďalej obsahuje tri zobrazenia aktuálnej teploty v úli, vlhkosti a váhy úľa. A tlačidlo, ktoré slúži na zobrazenie fragmentu Graph.



Obrázok 14 Monitoring Fragment

2.3.1.7 Graph Fragment



Obrázok 15 Graph Fragment

Fragment slúži na zobrazenie štatistík Teploty v úli, vlhkosti a váhy úľa grafickým spôsobom. Na výber sú 3 možnosti a to:

1. Za jeden deň,
2. Za mesiac,
3. Za celý čas.

Na zobrazenie grafov je využívaný GraphView.

2.3.2 Broadcast receiver

BroadcastReceiver je zo spôsobov, ako reagovať na správy zaslané inými aplikáciami alebo systémom.

V tomto konkrétnom prípade ide o BroadcastReceiver na zachytenie správy o upozornení zo systému. Využíva sa v triede LocalNotification, ktorá predstavuje vytvorenie notifikácie v určitý čas. Notifikácia je vytvorená v metóde onReceive Táto trieda je volaná cez AlarmManager.

```
class LocalNotification : BroadcastReceiver() {  
  
    override fun onReceive(context: Context, intent: Intent) {  
        createNotificationChannel(context)  
        val task = intent.getStringExtra("name: 1")  
        val notification = NotificationCompat.Builder(context, CHANNEL_ID)  
            .setContentTitle("Task notification")  
            .setContentText("$task")  
            .setSmallIcon(R.drawable.icon)  
            .setPriority(NotificationCompat.PRIORITY_HIGH)  
            .build()  
        val notificationManager = NotificationManagerCompat.from(context)  
        notificationManager.notify(NOTIFICATION_ID, notification)  
    }  
}
```

Obrázok 16 Ukážka z kódu využitia BroadCast receiver

2.3.3 DataBinding

Data Binding je knižnica a súčasne architektonický prvok v Android Studiu, ktorý umožňuje spojiť dáta medzi logikou aplikácie a užívateľským rozhraním pomocou deklaratívneho XML syntaxe. Čiže Data Binding umožňuje pripojiť kľúčové slová v XML súboroch priamo k dátovým modelom a výrazom v kóde.

V tejto semestrálnej práci sa využíva v každom Fragmente a každej triede, ktorá ma aj vlastné grafické rozhranie napr. vo vyskakovacích oknách.

```
val popupBinding = PopupLayoutBinding.inflate(LayoutInflater.from(context))
val titleEdit = popupBinding.titleEdit
val addressEdit = popupBinding.addressEdit

popupBinding.popupButton.setOnClickListener { it: View!
    val title = titleEdit.text.toString()
    val address = addressEdit.text.toString()
    onClick(title, address)
    this.popupWindow.dismiss()
}
```

Obrázok 17 Ukážka kódu využitia dataBindingu

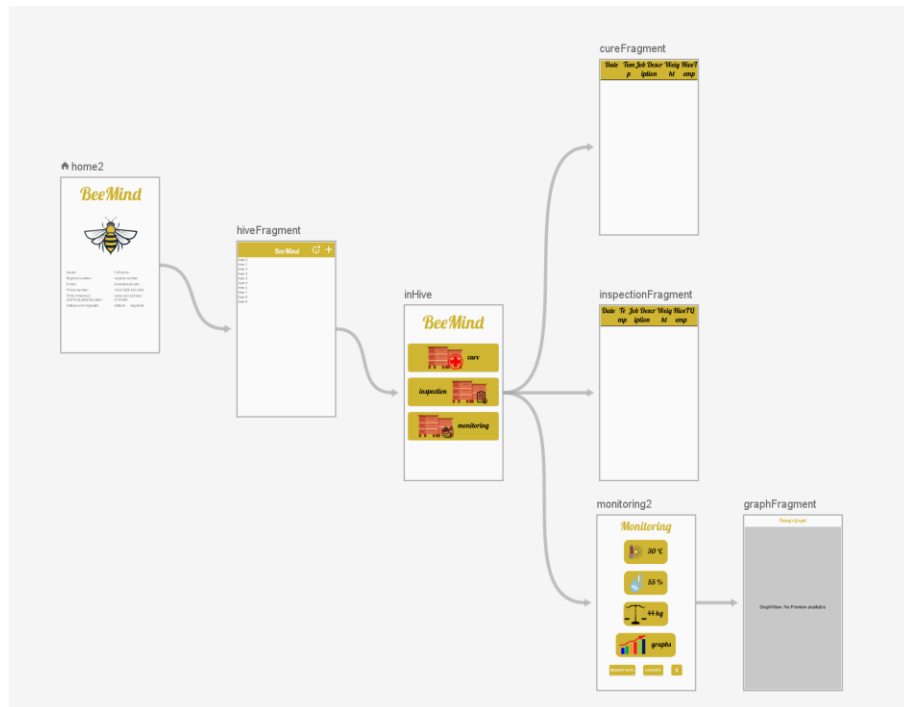
2.3.4 Navigation

Navigation sa odvoláva na interakcie, ktoré umožňujú používateľom navigovanie cez, dovnútra a naspäť z rôznych častí obsahu v aplikácii.

V tejto semestrálnej práci sa využíva v prechodoch medzi fragmentmi.

```
binding.cure.setOnClickListener { it: View!
    findNavController().navigate(R.id.action_inHive_to_cureFragment, bundle)
}
binding.inspection.setOnClickListener { it: View!
    findNavController().navigate(R.id.action_inHive_to_inspectionFragment, bundle)
}
binding.monitoring.setOnClickListener { it: View!
    findNavController().navigate(R.id.action_inHive_to_monitoring2, bundle)
}
```

Obrázok 18 Využitie Navigation v aplikácii



Obrázok 19 Navigation graph aplikácie

2.3.5 AlarmManager

Táto trieda sprostredkováva prístup k systémovým službám pre plánované zapnutie aplikácie alebo vykonanie určitej úlohy.

Konkrétne vytvorenie a zobrazenie notifikácie, ktorá slúži ako pripomienka pre včelára. AlarmManager je nastavený pri naplánovaní úlohy a je vytvorený jednorazovo.

```

val alarmManager =
    context.getSystemService(Context.ALARM_SERVICE) as AlarmManager
val intent = Intent(context, LocalNotification::class.java)
intent.putExtra(name: "1", value: "${taskEdit.text}")
val pendingIntent = PendingIntent.getBroadcast(
    context,
    requestCode: 0,
    intent,
    PendingIntent.FLAG_MUTABLE
)
alarmManager.setExact(AlarmManager.RTC_WAKEUP, alarmTime, pendingIntent)
  
```

Obrázok 20 Ukážka kódu pre vytvorenie AlarmManagera

2.3.6 Widget

V tejto semestrálnej práci slúži na zobrazenie dní od posledného liečenia a od poslednej inšpekcie.



2.3.7 Notifikácie

V tejto semestrálnej práci sa využívajú ako pripomienky pre včelára, kde:

1. Notifikácia sa zobrazí v čas zvolený používateľom.
2. Notifikácia, ktorá sa zobrazí pri zapnutí aplikácie na mieste včelnice a zobrazí úlohy, ktoré si jej používateľ naplánoval vykonať.

2.3.8 Dynamické rozloženie

Konkrétne v tejto aplikácii sa využíva v [Hive Fragmente](#) vo forme RecyclerView s adaptérom. Kde po pridaní nového úľa sa zobrazí v zozname novo pridaný úl.

A v [Cure Fragmente](#) a v [Inspection Fragmente](#), kde po pridaní záznamu sa tabuľka rozšíri o novo pridaný riadok záznamu.

2.3.9 GPS

V tejto aplikácii sa využíva na zistenie polohy užívateľa, kvôli notifikácií, ktorá sa zobrazí práve vtedy keď sa užívateľ nachádza na včelnici.

```
apiaryLocation.latitude = cursor.getDouble(latIndex)
apiaryLocation.longitude = cursor.getDouble(lonIndex)

val locationClient = LocationServices.getFusedLocationProviderClient(applicationContext)
locationClient.lastLocation.addOnSuccessListener { location: Location? ->
    if (location != null) {
        val distance = location.distanceTo(apiaryLocation) / 1000
        Toast.makeText(applicationContext, text: "distance is $distance", Toast.LENGTH_LONG).show()

        if (distance < 0.02) {
            notificationManager.notify(id: 1, builder.build())
            Tasks.delete(context: this)
        }
    }
}
```

Obrázok 21 Ukážka kódu pre získanie polohy a jej využitie v aplikácii

2.3.10 Bluetooth

Bluetooth sa v tejto aplikácii používa na komunikáciu užívateľa s riadiacou jednotkou nachádzajúcou sa v úli. Aplikácia posiela požiadavky a riadiaca jednotka jej obratom žiadané informácie posiela. Na prijímanie dát z riadiacej jednotky sa používa inputStream nachádzajúci sa v GlobalScope.

2.3.11 Štýly

V aplikácii sa nachádzajú dve vlastné štýly a to pre tlačidlo a pre rozloženie. Štýl nastavuje farbu pozadia tvar a zaoblenie okrajov.