ГУАП

КАФЕДРА № 42

ОТЧЕТ
ЗАЩИЩЕН С ОЦЕНКОЙ _____

ПРЕПОДАВАТЕЛЬ

старший преподаватель				С. Ю. Гуков
должность, уч. степень, звание		подпись, дата		инициалы, фамилия

ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №3

Намерения, меню и работа с данными

Вариант 5

по курсу: Разработка мобильных приложений. Разработка мобильных приложений на Kotlin

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ ГР. №	4128				В. А. Воробьев
			подпись, дата		инициалы, фамилия

Санкт-Петербург 2024

СОДЕРЖАНИЕ

1 Постановка задачи

1.1 Цель работы

Создайте приложение, которое получает данные с открытого API (например данные о погоде) и отображает их пользователю в удобном формате. Обеспечьте обработку ошибок при отсутствии интернет-соединения.

Задание «Фрагменты»:

- Создание Navigation Drawer или Bottom Navigation в соответствии с вариантом (предметной областью) – для определенного класса.
- Реализовать навигацию по пунктам меню.
- Создать фрагмент.
- Обработка переходов между фрагментами.
- Создать список с использованием RecycleView.

2 Выполнение работы

Не было найдено хорошего и бесплатного АПИ для варианта “Геометрические фигуры”, поэтому было принято решение взять BoredApi (URI - `http://www.boredapi.com`).

Для выполнения работы был выбран фреймворк `Jetpack Compose` в виду простоты его использования.

Итоговая структура проекта изображена на рисунке 2.1

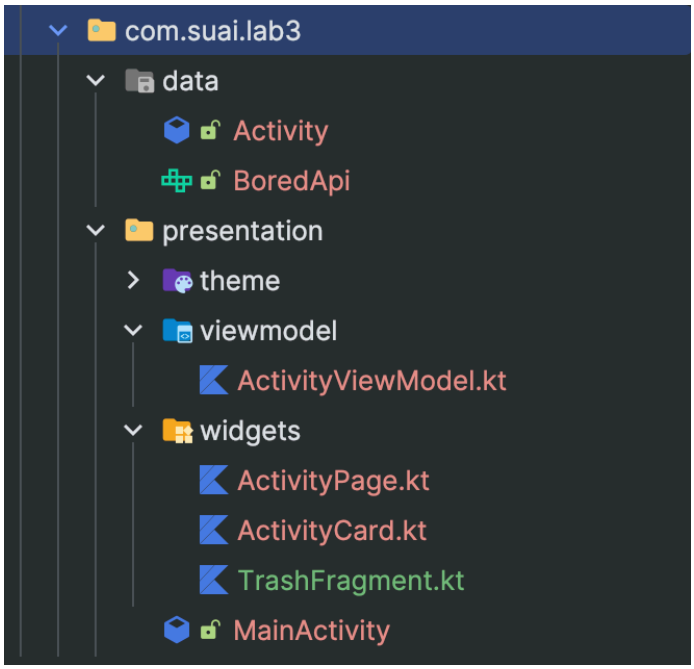

Рисунок 2.1 - Структура проекта

Код итогового проект доступен в Приложении и на GitHub (URI - `https://github.com/vladcto/suai-labs/tree/main/6_semester/ИТМ/lab3`).

Демонстрация работы представлена на рисунках 2.2 - 2.4. При ошибках возвращается пустой список и появляется белый экран.

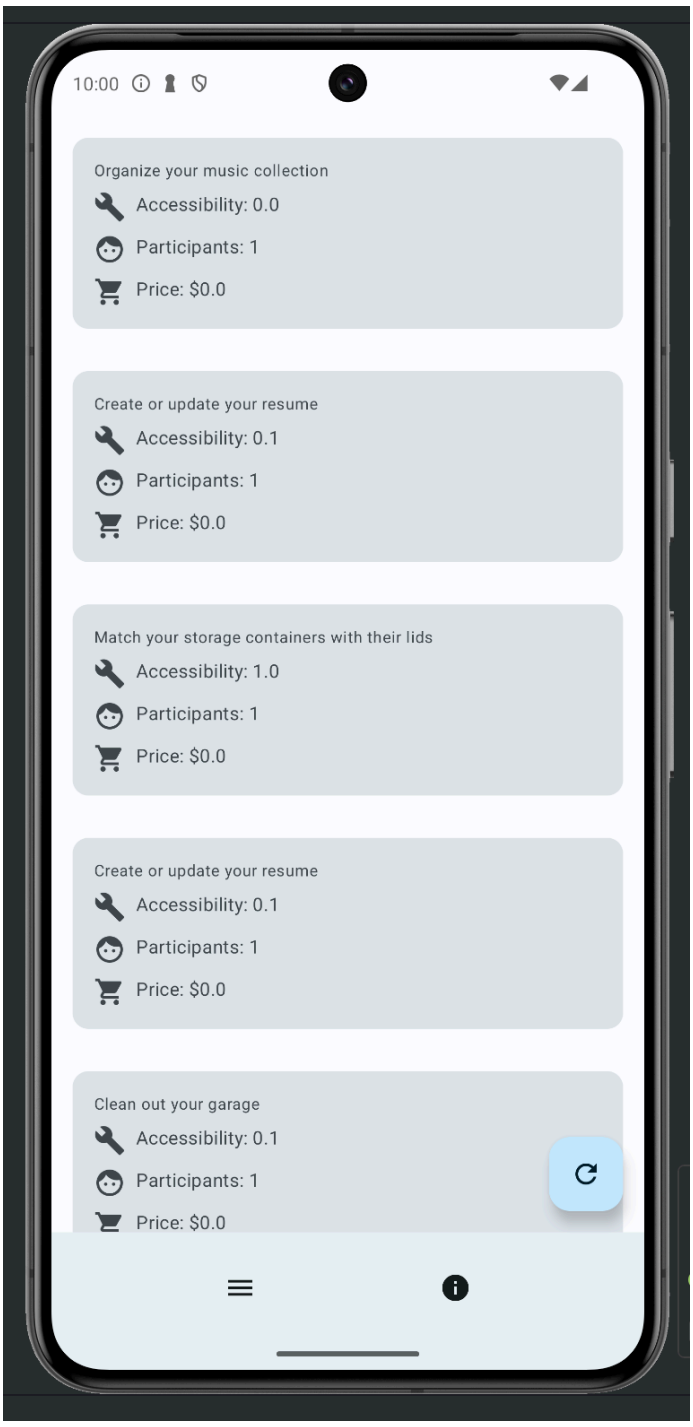

Рисунок 2.2 - Начальная страница

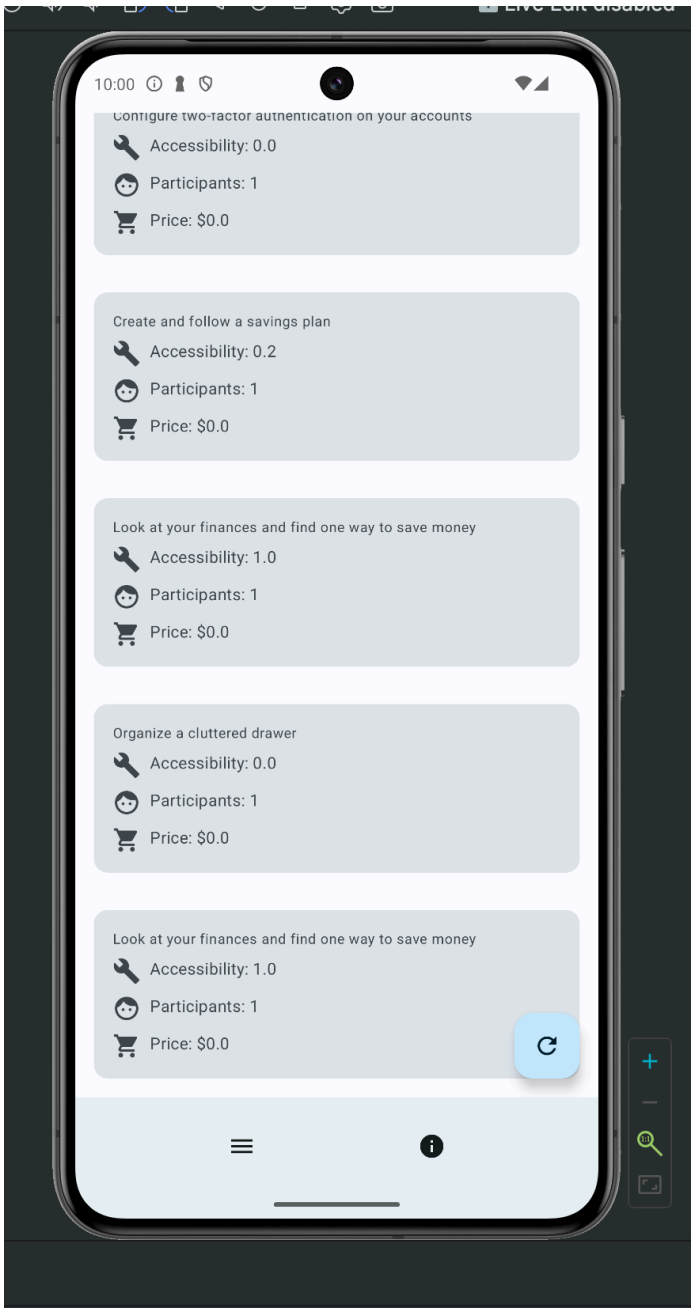

Рисунок 2.3 - Скролл страницы вниз

Рисунок 2.4 - Ошибка сети

3 Вывод

В результате выполнения лабораторной работы было успешно разработано мобильное приложение под ОС Android на языке программирования Kotlin с использованием фреймворка Jetpack Compose. Приложение включает в себя функционал для навигации посредством BottomBar, подгрузки апи и реагирования на ошибки.

Выполненная лабораторная работа способствовала углублению знаний и навыков в области разработки мобильных приложений.

ПРИЛОЖЕНИЕ

```
1  package com.suai.lab3.presentation.widgets
2
3  import androidx.compose.foundation.layout.Column
4  import androidx.compose.foundation.layout.Row
5  import androidx.compose.foundation.layout.Spacer
6  import androidx.compose.foundation.layout.fillMaxWidth
7  import androidx.compose.foundation.layout.height
8  import androidx.compose.foundation.layout.padding
9  import androidx.compose.foundation.layout.size
10 import androidx.compose.foundation.layout.width
11 import androidx.compose.material.icons.Icons
12 import androidx.compose.material.icons.filled.Build
13 import androidx.compose.material.icons.filled.Face
14 import androidx.compose.material.icons.filled.ShoppingCart
15 import androidx.compose.material3.Card
16 import androidx.compose.material3.Icon
17 import androidx.compose.material3.MaterialTheme
18 import androidx.compose.material3.Text
19 import androidx.compose.runtime.Composable
20 import androidx.compose.ui.Modifier
21 import androidx.compose.ui.unit.dp
22 import com.suai.lab3.data.Activity
23
24 @Composable
25 fun ActivityCard(activity: Activity) {
26     Card(
27         modifier = Modifier
28             .fillMaxWidth()
29             .padding(16.dp),
30     ) {
31         Column(
32             modifier = Modifier
33                 .padding(16.dp)
34                 .fillMaxWidth()
35         ) {
36             Text(
37                 text = activity.activity,
38                 style = MaterialTheme.typography.bodySmall,
39                 modifier = Modifier.fillMaxWidth()
40             )
```

```
41              Spacer(modifier = Modifier.height(8.dp))
42              Row {
43                  Icon(
44                      imageVector = Icons.Default.Build,
45                      contentDescription = "Accessibility",
46                      modifier = Modifier.size(24.dp)
47                  )
48                  Spacer(modifier = Modifier.width(8.dp))
49                  Text(
50                      text = "Accessibility: ${activity.
                            accessibility}",
51                      style = MaterialTheme.typography.
                            bodyMedium
52                  )
53              }
54              Spacer(modifier = Modifier.height(8.dp))
55              Row {
56                  Icon(
57                      imageVector = Icons.Default.Face,
58                      contentDescription = "Participants",
59                      modifier = Modifier.size(24.dp)
60                  )
61                  Spacer(modifier = Modifier.width(8.dp))
62                  Text(
63                      text = "Participants: ${activity.
                            participants}",
64                      style = MaterialTheme.typography.
                            bodyMedium
65                  )
66              }
67              Spacer(modifier = Modifier.height(8.dp))
68              Row {
69                  Icon(
70                      imageVector = Icons.Default.ShoppingCart,
71                      contentDescription = "Price",
72                      modifier = Modifier.size(24.dp)
73                  )
74                  Spacer(modifier = Modifier.width(8.dp))
75                  Text(
76                      text = "Price: $${activity.price}",
77                      style = MaterialTheme.typography.
```

```
                        bodyMedium
                    )
                }
            }
        }
}

package com.suai.lab3.presentation.widgets

import androidx.compose.foundation.lazy.LazyColumn
import androidx.compose.foundation.lazy.itemsIndexed
import androidx.compose.runtime.Composable
import androidx.compose.runtime.getValue
import androidx.compose.runtime.livedata.observeAsState
import androidx.compose.ui.Modifier
import com.suai.lab3.presentation.viewmodel.ActivityViewModel

@Composable
fun ActivityPage(viewModel: ActivityViewModel, modifier:
    Modifier) {
      val activities by viewModel.activities.observeAsState()
      LazyColumn(modifier = modifier) {
          itemsIndexed(activities ?: listOf()) { _, item ->
              ActivityCard(item)
          }
      }
}

package com.suai.lab3.presentation.viewmodel

import androidx.lifecycle.LiveData
import androidx.lifecycle.MutableLiveData
import androidx.lifecycle.ViewModel
import androidx.lifecycle.ViewModelProvider
import androidx.lifecycle.viewModelScope
import com.suai.lab3.data.Activity
import com.suai.lab3.data.BoredApi
import kotlinx.coroutines.CoroutineExceptionHandler
import kotlinx.coroutines.Dispatchers
import kotlinx.coroutines.launch
import kotlinx.coroutines.withContext
```

```
117 import retrofit2.Retrofit
118 import retrofit2.await
119 import retrofit2.converter.gson.GsonConverterFactory
120
121 val coroutineExceptionHandler = CoroutineExceptionHandler { _
      , throwable ->
122     throwable.printStackTrace()
123 }
124
125 class ActivityViewModel(private val api: BoredApi) :
      ViewModel() {
126     private val _activities = MutableLiveData<List<Activity
          >>()
127     val activities: LiveData<List<Activity>> = _activities
128
129     fun getActivity() {
130         viewModelScope.launch(Dispatchers.IO +
              coroutineExceptionHandler) {
131             val result = mutableListOf<Activity>()
132             (0..20).forEach { _ ->
133                 try{
134                     val activity = api.getActivity("busywork
                          ").await()
135                 }catch(_: Throwable){}
136                 result.add(activity)
137             }
138             withContext(Dispatchers.Main +
                  coroutineExceptionHandler) {
139                 _activities.postValue(result)
140             }
141         }
142     }
143
144     companion object {
145         val Factory: ViewModelProvider.Factory = object :
              ViewModelProvider.Factory {
146             @Suppress("UNCHECKED_CAST")
147             override fun <T : ViewModel> create(modelClass:
                  Class<T>): T {
148                 // Ну и ладно, что не Dagger
149                 val retrofit = Retrofit.Builder()
```

```
150                         .baseUrl("https://boredapi.com/api/")
151                         .addConverterFactory(GsonConverterFactory
                                .create())
152                         .build()
153
154                     return ActivityViewModel(
155                         retrofit.create(BoredApi::class.java)
156                     ) as T
157                 }
158             }
159         }
160 }
161
162 package com.suai.lab3.data
163
164 data class Activity(
165     val activity: String,
166     val accessibility: Double,
167     val participants: Int,
168     val price: Double,
169 )
170
171 package com.suai.lab3.data
172
173 import retrofit2.Call
174 import retrofit2.http.GET
175 import retrofit2.http.Query
176
177 interface BoredApi {
178     @GET("activity")
179     fun getActivity(@Query("type") type: String): Call<
            Activity>
180 }
```