

Министерство образования и молодежной политики Свердловской области

ГАПОУ СО «Екатеринбургский колледж транспортного строительства»

Отчёт по учебной практике

Задание 2. Приложение - умные часы

Вариант 5

Выполнил: Харсевич С.Л.

Группа: ПР-31

Преподаватель: Мирошниченко Г.В.

2023

## **Описание задачи**

Разработать мобильное приложение – умные часы, информация в приложении

должна сохраняться в SharedPreferences, в формате JSON. Приложение должно

содержать 4 экрана:

- приветственный экран с авторизацией пользователя, проверкой

ввода на пустоту и сохранением информации о пользователе через механизм

SharedPreferences.

- содержать форму ввода данных

- экран, содержащий меню со значками

- список из меню с использованием библиотеки ROOM,

возможностью добавления, удаления введенных данных, с сохранением в базе

данных (ROOM или SQL).

- стилизация приложения в соответствии с вариантом

- изменить название, тему и иконку приложения (должны

соответствовать названию)

- предусмотреть портретную ориентацию

Приложение «Комфортное путешествие»

**Листинг**

MainActivity

<androidx.wear.widget.BoxInsetLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:padding="@dimen/box\_inset\_layout\_padding"  
 tools:context=".MainActivity"  
 tools:deviceIds="wear"  
 android:background="@color/back">  
  
 <FrameLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 app:layout\_boxedEdges="all">  
  
 <TextView  
 android:textAlignment="center"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="30dp"  
 android:layout\_marginLeft="10dp"  
 android:layout\_marginTop="10dp"  
 android:text="@string/app"  
 android:textColor="@color/text"  
 android:textSize="13dp"  
 android:textStyle="bold" />  
  
 <TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="@string/email"  
 android:textSize="12dp"  
 android:layout\_marginLeft="10dp"  
 android:textColor="@color/text"  
 android:layout\_marginTop="35dp"/>  
 <EditText  
 android:inputType="textEmailAddress"  
 android:id="@+id/email"  
 android:layout\_width="160dp"  
 android:textColorHint="@color/hint"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="50dp"  
 android:layout\_marginLeft="10dp"  
 android:textSize="12dp"  
 android:hint="@string/email\_hint"/>  
 <TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="@string/password"  
 android:textSize="12dp"  
 android:layout\_marginLeft="10dp"  
 android:textColor="@color/text"  
 android:layout\_marginTop="85dp"/>  
 <EditText  
 android:inputType="textPassword"  
 android:id="@+id/pass"  
 android:layout\_width="160dp"  
 android:textColorHint="@color/hint"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="100dp"  
 android:layout\_marginLeft="10dp"  
 android:textSize="12dp"  
 android:hint="@string/writePas"/>  
  
 <Button  
 android:onClick="perehod"  
 android:id="@+id/singin"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:layout\_marginTop="140dp"  
 android:text="@string/log"  
 android:textSize="12dp"  
 android:textStyle="bold"  
 android:textAllCaps="false"  
 android:background="@color/text"/>  
 </FrameLayout>  
</androidx.wear.widget.BoxInsetLayout>

import android.app.Activity  
import android.content.Intent  
import android.content.SharedPreferences  
import android.os.Bundle  
import android.view.View  
import android.widget.EditText  
import android.widget.Toast  
import com.example.komfort.databinding.ActivityMainBinding  
  
class MainActivity : Activity() {  
  
 private lateinit var binding: ActivityMainBinding  
  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
  
 binding = ActivityMainBinding.inflate(*layoutInflater*)  
 setContentView(binding.*root*)  
 val sharedPreferences = getSharedPreferences("LoginAndPassword", *MODE\_PRIVATE*)  
 val editor: SharedPreferences.Editor = sharedPreferences.edit()  
  
 editor.putString("username", "ects")  
 editor.putString("password", "ects")  
 editor.apply()  
  
  
 }  
 fun perehod(view: View) {  
  
 val usernameEditText: EditText = findViewById(R.id.*email*)  
 val passwordEditText: EditText = findViewById(R.id.*pass*)  
  
 val log :String = usernameEditText.*text*.toString()  
 val pas:String = passwordEditText.*text*.toString()  
  
  
 if (usernameEditText.*text*.toString().*isEmpty*() || passwordEditText.*text*.toString().*isEmpty*()) {  
  
 Toast.makeText(this@MainActivity, "Введите логин и пароль", Toast.*LENGTH\_SHORT*).show()  
 }  
 else {  
  
 val sharedPreferences = getSharedPreferences("LoginAndPassword", *MODE\_PRIVATE*)  
  
 val savedUsername = sharedPreferences.getString("username", "")  
 val savedPassword = sharedPreferences.getString("password", "")  
  
  
 if (log == savedUsername && pas == savedPassword) {  
 val intent = Intent(this@MainActivity, MainScreen::class.*java*)  
 startActivity(intent)  
 } else {  
 Toast.makeText(this@MainActivity, "Неверный логин или пароль", Toast.*LENGTH\_SHORT*).show()  
  
 }  
  
 }  
  
  
 }  
}

MainScreen

<androidx.wear.widget.BoxInsetLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:padding="@dimen/box\_inset\_layout\_padding"  
 tools:context=".MainScreen"  
 tools:deviceIds="wear"  
 tools:ignore="MissingDefaultResource"  
 android:background="@color/back">  
  
 <FrameLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:padding="@dimen/inner\_frame\_layout\_padding"  
 app:layout\_boxedEdges="all">  
 <LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:padding="9dp"  
 android:orientation="vertical">  
  
 <ImageButton  
 android:id="@+id/discussionsImageView"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:background="@drawable/img\_2"/>  
  
 <TextView  
 android:layout\_marginStart="16dp"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:textAlignment="center"  
 android:text="@string/speak"  
 android:textSize="12sp"  
 android:textColor="@color/text"  
 tools:ignore="SmallSp" />  
  
 </LinearLayout>  
  
 <LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:layout\_marginLeft="90dp"  
 android:orientation="vertical"  
 android:layout\_marginTop="9dp">  
  
 <ImageButton  
 android:id="@+id/setsImageView"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="74dp"  
 android:layout\_marginEnd="5dp"  
 android:background="@drawable/img\_3"/>  
  
 <TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:textAlignment="center"  
 android:text="@string/speak\_2"  
 android:textSize="12sp"  
 android:layout\_marginStart="4dp"  
 android:textColor="@color/text"  
 tools:ignore="SmallSp" />  
  
 </LinearLayout>  
 </FrameLayout>  
</androidx.wear.widget.BoxInsetLayout>

import android.app.Activity  
import android.content.Intent  
import android.os.Bundle  
import android.widget.ImageButton  
import com.example.komfort.databinding.ActivityMainScreenBinding  
  
class MainScreen : Activity() {  
 private lateinit var speak: ImageButton  
 private lateinit var speak2: ImageButton  
 private lateinit var binding: ActivityMainScreenBinding  
  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
  
 binding = ActivityMainScreenBinding.inflate(*layoutInflater*)  
 setContentView(binding.*root*)  
 speak=findViewById(R.id.*discussionsImageView*)  
 speak2=findViewById(R.id.*setsImageView*)  
 speak.setOnClickListener**{** val intent= Intent(this,Search\_map::class.*java*)  
 startActivity(intent)  
 **}** speak2.setOnClickListener **{** val intent= Intent(this, List\_screen::class.*java*)  
 startActivity(intent)  
 **}** }  
}

Search\_map

<androidx.wear.widget.BoxInsetLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:padding="@dimen/box\_inset\_layout\_padding"  
 tools:context=".Search\_map"  
 tools:deviceIds="wear"  
 android:background="@color/back">  
<ScrollView  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 app:layout\_boxedEdges="all">  
  
 <LinearLayout  
 android:orientation="vertical"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content">  
 <EditText  
 android:textColor="@color/text"  
 android:textColorHint="@color/text"  
 android:id="@+id/findSity"  
 android:layout\_width="160dp"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginLeft="10dp"  
 android:textSize="12dp"  
 android:hint="Введите город"/>  
 <Button  
 android:onClick="SearhButton"  
 android:id="@+id/searchBut"  
 android:text="@string/search"  
 android:background="@color/text"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent" />  
  
 <ImageView  
 android:adjustViewBounds="true"  
 android:scaleType="centerCrop"  
 android:id="@+id/imgKarta"  
 android:layout\_marginTop="10dp"  
 android:layout\_width="match\_parent"  
 android:layout\_height="100dp" />  
  
 <TextView  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:id="@+id/text\_info" />  
 </LinearLayout>  
</ScrollView>  
  
</androidx.wear.widget.BoxInsetLayout>

import android.annotation.SuppressLint  
import android.app.Activity  
import android.content.SharedPreferences  
import android.os.Bundle  
import android.util.Log  
import android.view.View  
import android.widget.Button  
import android.widget.EditText  
import android.widget.ImageView  
import android.widget.TextView  
import com.android.volley.Request  
import com.android.volley.RequestQueue  
import com.android.volley.Response  
import com.android.volley.toolbox.JsonObjectRequest  
import com.android.volley.toolbox.StringRequest  
import com.android.volley.toolbox.Volley  
import com.google.android.gms.maps.model.LatLng  
import com.example.komfort.databinding.ActivitySearchMapBinding  
import com.google.maps.android.SphericalUtil  
import com.squareup.picasso.Picasso  
import org.json.JSONArray  
import org.json.JSONException  
import org.json.JSONObject  
  
class Search\_map : Activity() {  
private lateinit var editText: EditText  
private lateinit var Image\_map:ImageView  
private lateinit var info\_sity:TextView  
 private lateinit var binding: ActivitySearchMapBinding  
  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
  
 binding = ActivitySearchMapBinding.inflate(*layoutInflater*)  
 setContentView(binding.*root*)  
 Image\_map = findViewById(R.id.*imgKarta*)  
 editText = findViewById(R.id.*findSity*)  
 info\_sity=findViewById(R.id.*text\_info*)  
 val sharedPreferences = getSharedPreferences("loc", *MODE\_PRIVATE*)  
 val editor: SharedPreferences.Editor = sharedPreferences.edit()  
  
 editor.putString("a", "60.597474")  
 editor.putString("b", "56.838011")  
 editor.apply()  
  
 }  
 fun SearhButton(view: View){  
 zapros(editText.*text*.toString())  
 }  
 private fun zapros(sity: String){  
 Log.d("MyLog", "zapros() method called")  
  
 // Проверка наличия разрешения на использование интернета  
  
  
 val url = "https://geocode-maps.yandex.ru/1.x/?apikey=3ee7e538-e94a-42c4-ac69-00f6160dfd34&geocode=$sity&format=json"  
  
 val queue = Volley.newRequestQueue(this)  
  
 val stringRequest = StringRequest(  
 Request.Method.*GET*, url,  
 **{** response **->** // Обработка успешного ответа  
 try {  
 val obj = JSONObject(response)  
 val featureMember = obj.getJSONObject("response")  
 .getJSONObject("GeoObjectCollection")  
 .getJSONArray("featureMember")  
  
 if (featureMember.length() > 0) {  
 val firstObject = featureMember.getJSONObject(0)  
 val point = firstObject.getJSONObject("GeoObject")  
 .getJSONObject("Point")  
 .getString("pos")  
 val tochki = point.toString().*split*(" ")  
 //txt.text = "${tochki[0]}UU${tochki[1]} "  
 searchKarta(tochki,sity)  
  
 Log.d("MyLog", "Coordinates: $point")  
 } else {  
 Log.d("MyLog", "No features found in the response")  
 }  
 } catch (e: JSONException) {  
 Log.d("MyLog", "JSON parsing error: ${e.message}")  
 }  
 **}**,  
 **{** error **->** // Обработка ошибки  
 val statusCode = error.networkResponse.statusCode  
 Log.d("MyLog", "Volley error status code: $statusCode")  
  
  
 **}**)  
  
 // Добавление запроса в очередь  
 queue.add(stringRequest)  
 }  
 private fun zapros\_info(sity: String,callback: (String) -> Unit){  
 Log.d("MyLog", "zapros\_info() method called")  
  
 val url =  
 "https://search-maps.yandex.ru/v1/?text=Достопримечательности%20$sity&type=biz&lang=ru\_RU&apikey=5ff8c278-65fd-4803-b43e-ff99ff9b21d1"  
  
 val requestQueue = Volley.newRequestQueue(this)  
  
 val request = JsonObjectRequest(Request.Method.*GET*, url, null,  
 **{** response **->** try {  
 val features: JSONArray = response.getJSONArray("features")  
 val stringBuilder = StringBuilder()  
  
 for (i in 0 *until* features.length()) {  
 val feature: JSONObject = features.getJSONObject(i)  
 val properties: JSONObject = feature.getJSONObject("properties")  
 val name: String? = properties.optString("name", "")  
 val info: String = "$sity, $name"  
 stringBuilder.append(info).append("\n")  
  
  
 Log.d("MyLog", "Город $sity Достопримечательность $name")  
  
 val db = MainDb.getDb(this)  
 val item = Item(null, sity.toString(), name.*toString*())  
 Thread**{** db.getDao().insertItem(item)  
 **}**.start()  
 }  
  
 callback(stringBuilder.toString())  
 } catch (e: JSONException) {  
 callback("")  
 e.printStackTrace()  
 }  
 **}**,  
 **{** error **->** callback("")  
 error.printStackTrace()  
 **}**)  
 requestQueue.add(request)  
 }  
 fun searchKarta(array: List<String>,sity: String){  
 val imageUrl = "https://static-maps.yandex.ru/v1?ll=${array[0]},${array[1]}&size=450,450&z=13&pt=${array[0]},${array[1]},pmwtm1~${array[0]},${array[1]},pmwtm99&apikey=f9ce7b23-8786-44b7-8308-864c74bf640a"  
  
 Picasso.get().load(imageUrl).into(Image\_map)  
 val stringBuilder = StringBuilder()  
 val name = *mutableListOf*<String>()  
 zapros\_info(sity) **{** info **->** stringBuilder.append(info).append("\n") // Append the received info  
 info\_sity.append("$info\n") // Add each value to the EditText  
 **}** val info = stringBuilder.toString()  
 info\_sity.*text* = info // Set the combined info to the info\_sity TextView  
  
  
 }  
}

Static\_search

<androidx.wear.widget.BoxInsetLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:padding="@dimen/box\_inset\_layout\_padding"  
 tools:context=".Static\_search"  
 tools:deviceIds="wear"  
 android:background="@color/back">  
 <ScrollView  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent">  
  
 <LinearLayout  
 android:layout\_width="180dp"  
 android:layout\_height="wrap\_content"  
 android:orientation="vertical">  
 <LinearLayout  
 android:layout\_marginLeft="120dp"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:orientation="horizontal">  
 <ImageButton  
 android:scaleType="centerCrop"  
 android:id="@+id/editButton"  
 android:layout\_width="30dp"  
 android:layout\_height="30dp"  
 android:background="@drawable/img"></ImageButton>  
 <ImageButton  
 android:scaleType="centerCrop"  
 android:id="@+id/deleteButton"  
 android:layout\_width="30dp"  
 android:layout\_height="30dp"  
 android:background="@drawable/img\_1"></ImageButton>  
 </LinearLayout>  
  
 <TextView  
 android:textColor="@color/text"  
 android:id="@+id/titleTextView"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:textSize="18sp"  
 android:textStyle="bold"  
 android:paddingStart="16dp"  
 android:paddingTop="8dp"  
 android:paddingEnd="16dp"  
 android:paddingBottom="4dp" />  
  
 <TextView  
 android:textColor="@color/text"  
 android:id="@+id/descriptionTextView"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:textSize="14sp"  
 android:paddingStart="16dp"  
 android:paddingTop="4dp"  
 android:paddingEnd="16dp"  
 android:paddingBottom="8dp" />  
 </LinearLayout>  
 </ScrollView>  
</androidx.wear.widget.BoxInsetLayout>

List\_screen

<androidx.wear.widget.BoxInsetLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:padding="@dimen/box\_inset\_layout\_padding"  
 tools:context=".List\_screen"  
 tools:deviceIds="wear"  
 android:theme="@style/Theme.AppCompat"  
 android:background="@color/back">  
 <LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:orientation="horizontal">  
  
 <Button  
 android:background="@color/text"  
 android:id="@+id/clearButton"  
 android:layout\_width="0dp"  
 android:layout\_height="20dp"  
 android:layout\_weight="1"  
 android:text="очистить"  
 android:textColor="@color/white"  
 android:layout\_margin="4dp"  
 android:onClick="ButCleaAll"/>  
  
  
 <Button  
 android:textColor="@color/white"  
 android:background="@color/text"  
 android:id="@+id/AddButton"  
 android:layout\_width="0dp"  
 android:layout\_height="20dp"  
 android:layout\_weight="1"  
 android:text="добавить"  
 android:layout\_margin="4dp"  
 android:onClick="AddNewButton"/>  
 </LinearLayout>  
 <ListView  
 android:id="@+id/listView"  
 android:layout\_marginTop="30dp"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent" />  
</androidx.wear.widget.BoxInsetLayout>

import android.app.Activity  
import android.app.AlertDialog  
import android.os.Bundle  
import android.view.View  
import android.widget.EditText  
import android.widget.LinearLayout  
import android.widget.ListView  
import androidx.appcompat.app.AppCompatActivity  
import androidx.lifecycle.asLiveData  
import com.example.komfort.databinding.ActivityListScreenBinding  
import kotlinx.coroutines.CoroutineScope  
import kotlinx.coroutines.Dispatchers  
import kotlinx.coroutines.launch  
import kotlinx.coroutines.withContext  
  
class List\_screen : AppCompatActivity() {  
 private val dataList = *mutableListOf*<Item>()  
  
 private lateinit var binding: ActivityListScreenBinding  
  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
  
 binding = ActivityListScreenBinding.inflate(*layoutInflater*)  
 setContentView(binding.*root*)  
 printInfo()  
  
 }  
 private fun printInfo() {  
 val textList = findViewById<ListView>(R.id.*listView*)  
 dataList.clear()  
  
 val db = MainDb.getDb(this)  
  
 db.getDao().getAllItem().*asLiveData*().observe(this) **{** list **->** val updatedDataList = *mutableListOf*<Item>()  
 list.*forEach* **{** item **->** val text = "id ${item.id}, Имя ${item.name\_sity} Профиль ${item.profile} \n"  
 // Создайте новый элемент Item и добавьте его в dataList  
 val newItem = Item(id = item.id, name\_sity = item.name\_sity, profile = item.profile)  
 updatedDataList.add(newItem)  
 **}** // После завершения цикла, обновите dataList  
 dataList.clear()  
 dataList.addAll(updatedDataList)  
  
 // Создайте или обновите адаптер  
 val itemAdapter = ItemAdapter(this, dataList,  
 onDeleteClick = **{** itemToDelete **->** // Здесь выполняйте удаление элемента из базы данных и обновление списка  
 val db = MainDb.getDb(this)  
 *CoroutineScope*(Dispatchers.IO).*launch* **{** try {  
 db.getDao().deleteItem(itemToDelete)  
 // После удаления, обновите список данных  
 printInfo()  
 } catch (e: Exception) {  
 // Обработка ошибки удаления, если произошла  
 }  
 **}  
 }**,  
 onEditClick = **{** position **->** // Получаю рание данные  
 val db = MainDb.getDb(this)  
 var bdNewText: Array<String> = *arrayOf*("id", "name\_sity", "profile")  
 *CoroutineScope*(Dispatchers.IO).*launch* **{** try {  
 bdNewText= *arrayOf*(position.id.*toString*(), position.name\_sity, position.profile)  
 } catch (e: Exception) {  
 // Обработка ошибки удаления, если произошла  
 }  
 **}** //изменения через AlertDiolog  
 val builder = AlertDialog.Builder(this)  
 builder.setTitle("Измените данные")  
  
 // Создайте контейнер LinearLayout для размещения двух EditText  
 val layout = LinearLayout(this)  
 layout.*orientation* = LinearLayout.*VERTICAL* // Создайте два EditText для ввода данных  
 val EditNumber = EditText(this)  
 val EditFakt = EditText(this)  
  
 // Установите хинты для EditText  
 EditNumber.*hint* = "Введите город"  
 EditFakt.*hint* = "Введите достопримечательность"  
 EditNumber.setText(bdNewText[1])  
 EditFakt.setText(bdNewText[2])  
 // Добавьте EditText к контейнеру  
 layout.addView(EditNumber)  
 layout.addView(EditFakt)  
  
 builder.setView(layout)  
  
 // Установите кнопку "OK" для сохранения данных  
 builder.setPositiveButton("OK") **{** dialog, \_ **->** var nameTxt = EditNumber.*text*.toString()  
 var profileTxt = EditFakt.*text*.toString()  
  
  
 //запись в бд  
 //проверка  
 if(nameTxt.*isEmpty*()){  
 nameTxt = bdNewText[2];  
 }  
 if(profileTxt.*isEmpty*()){  
 profileTxt = bdNewText[3];  
 }  
 //сама запись  
 val item = Item(bdNewText[0].*toInt*(), nameTxt, profileTxt)  
 *CoroutineScope*(Dispatchers.IO).*launch* **{** // Вызовите suspend-функцию updateItem в корутине  
 db.getDao().updateItem(item)  
 **}** printInfo()  
  
 **}** // Установите кнопку "Отмена" для закрытия диалога  
 builder.setNegativeButton("Отмена") **{** dialog, \_ **->** dialog.cancel()  
 **}** val dialog = builder.create()  
 dialog.show()  
 printInfo()  
  
 **}**)  
 textList.*adapter* = itemAdapter  
  
 // Уведомите адаптер о изменениях  
 itemAdapter.notifyDataSetChanged()  
  
 **}** }  
 fun AddNewButton(view: View) {  
 val builder = AlertDialog.Builder(this)  
 builder.setTitle("Введите данные")  
  
 // Создайте контейнер LinearLayout для размещения двух EditText  
 val layout = LinearLayout(this)  
 layout.*orientation* = LinearLayout.*VERTICAL* // Создайте два EditText для ввода данных  
 val EditNumber = EditText(this)  
 val EditFakt = EditText(this)  
  
 // Установите хинты для EditText  
 EditNumber.*hint* = "Введите город"  
 EditFakt.*hint* = "Введите достопримечательность"  
  
 // Добавьте EditText к контейнеру  
 layout.addView(EditNumber)  
 layout.addView(EditFakt)  
  
 builder.setView(layout)  
  
 // Установите кнопку "OK" для сохранения данных  
 builder.setPositiveButton("OK") **{** dialog, \_ **->** val nameTxt = EditNumber.*text*.toString()  
 val profileTxt = EditFakt.*text*.toString()  
  
 // Здесь можно обработать введенные данные (value1 и value2)  
 //запись в бд  
 if(nameTxt.length == 0 || profileTxt.length == 0){  
 val builder = AlertDialog.Builder(this)  
 builder.setTitle("Не сохранено")  
 .setMessage("Необходимо заполнить все поля")  
 .setPositiveButton("ОК") **{** dialog, id **->** dialog.cancel()  
 **}** builder.create()  
 }  
 else{  
 //сама запись  
 val db = MainDb.getDb(this)  
 val item = Item(null, nameTxt, profileTxt)  
 Thread**{** db.getDao().insertItem(item)  
 **}**.start()  
 dialog.dismiss()  
 printInfo()  
 }  
  
 **}** // Установите кнопку "Отмена" для закрытия диалога  
 builder.setNegativeButton("Отмена") **{** dialog, \_ **->** dialog.cancel()  
 **}** val dialog = builder.create()  
 dialog.show()  
  
 }  
 fun ButCleaAll(view: View) {  
 val db = MainDb.getDb(this)  
 *CoroutineScope*(Dispatchers.IO).*launch* **{** // Очистите все таблицы в базе данных  
 db.clearAllTables()  
  
 // После удаления данных, выполните операции обновления UI на главном потоке (если необходимо)  
 withContext(Dispatchers.Main) **{** // Обновление UI, если необходимо  
 **}  
 }** printInfo()  
 }  
}

class Item

import androidx.room.ColumnInfo  
import androidx.room.Entity  
import androidx.room.PrimaryKey  
@Entity (tableName = "items")  
data class Item (  
 //ид  
 @PrimaryKey(autoGenerate = true)  
 var id: Int? = null,  
  
 //Имя  
 @ColumnInfo(name = "name\_sity")  
 var name\_sity: String,  
  
 //Профиль  
 @ColumnInfo(name = "profile")  
 var profile: String,  
  
 )

interface Dao

import androidx.room.Dao  
import androidx.room.Delete  
import androidx.room.Insert  
import androidx.room.Query  
import androidx.room.Update  
import kotlinx.coroutines.flow.Flow  
  
@Dao  
interface Dao {  
 //запись  
 @Insert  
 fun insertItem(item: Item)  
  
 //получение всей базы  
 @Query("SELECT \* FROM items")  
 fun getAllItem(): Flow<List<Item>>  
  
 //удаление  
 @Delete  
 suspend fun deleteItem(item: Item)  
  
 //обновление  
  
 @Update  
 suspend fun updateItem(item: Item)  
}

class MainDb

import android.content.Context  
import androidx.room.Database  
import androidx.room.Room  
import androidx.room.RoomDatabase  
  
@Database(entities = [Item::class], version = 1)  
  
abstract class MainDb : RoomDatabase() {  
 abstract fun getDao():Dao  
  
 companion object{  
 fun getDb(context: Context): MainDb{  
 return Room.databaseBuilder(  
 context.*applicationContext*, MainDb::class.*java*, "test.db"  
 ).build()  
 }  
 }  
}

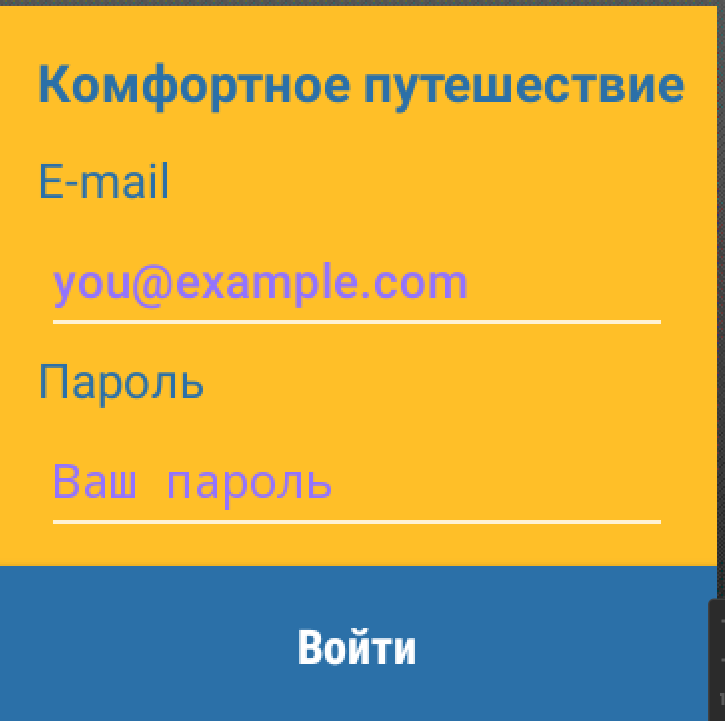
class ItemAdapter

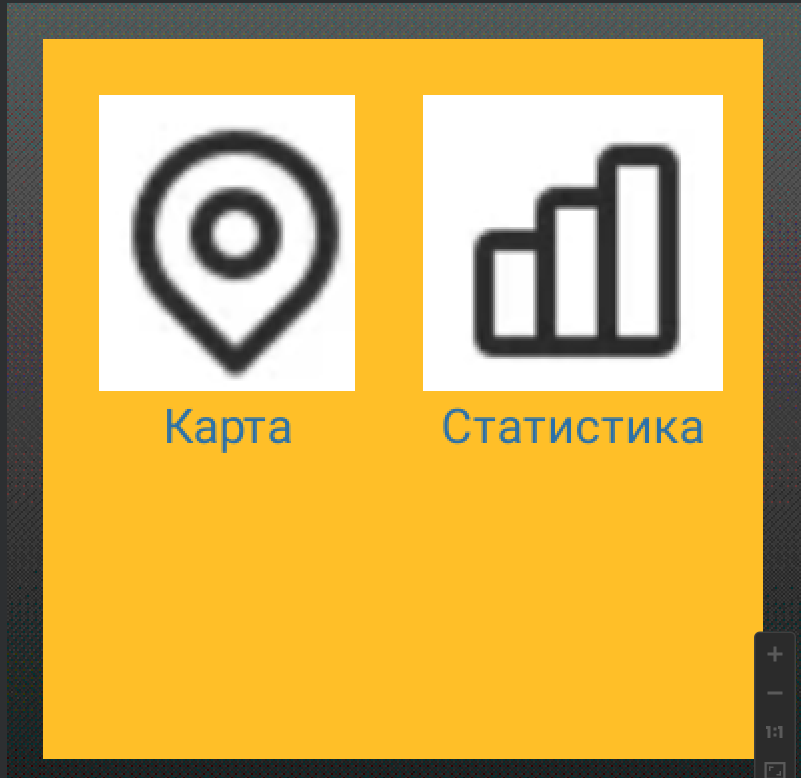
import android.content.Context  
import android.view.LayoutInflater  
import android.view.View  
import android.view.ViewGroup  
import android.widget.BaseAdapter  
import android.widget.ImageButton  
import android.widget.TextView  
  
class ItemAdapter(  
 context: Context,  
 private val itemList: List<Item>,  
 private val onDeleteClick: (Item) -> Unit,  
 private val onEditClick: (Item) -> Unit  
) : BaseAdapter() {  
 private val inflater: LayoutInflater = LayoutInflater.from(context)  
  
 override fun getCount(): Int {  
 return itemList.size  
 }  
  
 override fun getItem(position: Int): Any {  
 return itemList[position]  
 }  
  
 override fun getItemId(position: Int): Long {  
 return position.toLong()  
 }  
  
 override fun getView(position: Int, convertView: View?, parent: ViewGroup?): View {  
 val view: View  
 val holder: ViewHolder  
  
 if (convertView == null) {  
 view = inflater.inflate(R.layout.*activity\_static\_search*, parent, false)  
 holder = ViewHolder(  
 view.findViewById(R.id.*titleTextView*),  
 view.findViewById(R.id.*descriptionTextView*),  
 view.findViewById(R.id.*editButton*),  
 view.findViewById(R.id.*deleteButton*)  
 )  
 view.*tag* = holder  
 } else {  
 view = convertView  
 holder = view.*tag* as ViewHolder  
 }  
  
 val item = itemList[position]  
 holder.titleTextView.*text* = item.name\_sity // Отображаем "Имя" в заголовке  
 holder.descriptionTextView.*text* = item.profile // отображаем профиль  
  
 // Обработка нажатий на кнопку "Редактировать"  
 holder.editButton.setOnClickListener **{** onEditClick(item)  
 **}** // Обработка нажатий на кнопку "Удалить"  
 holder.deleteButton.setOnClickListener **{** if (itemList.*isNotEmpty*()) {  
 onDeleteClick(item)  
 }  
 **}** return view  
 }  
  
 private class ViewHolder(  
 val titleTextView: TextView,  
 val descriptionTextView: TextView,  
 val editButton: ImageButton,  
 val deleteButton: ImageButton  
 )  
}

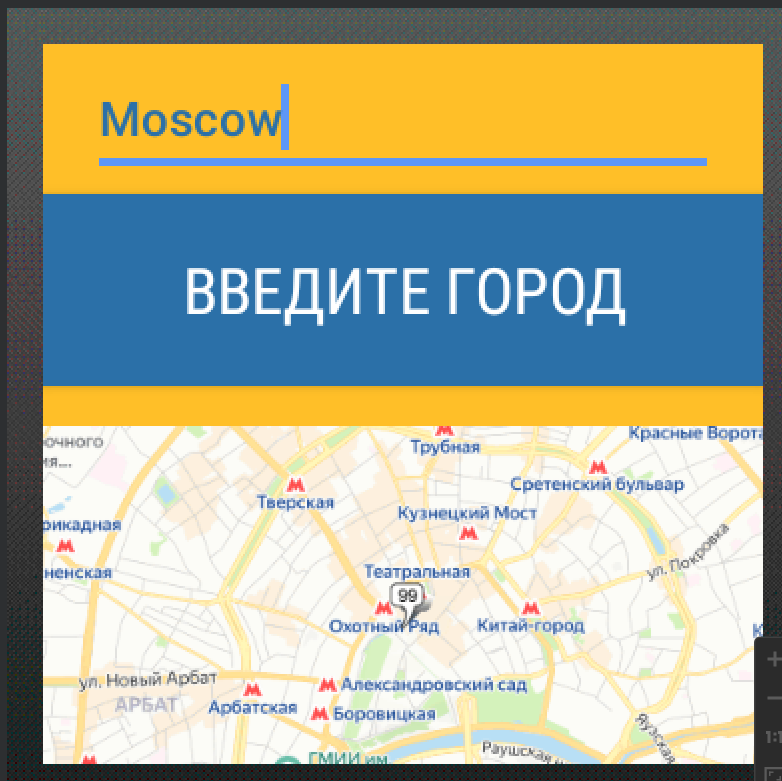
Библиотеки

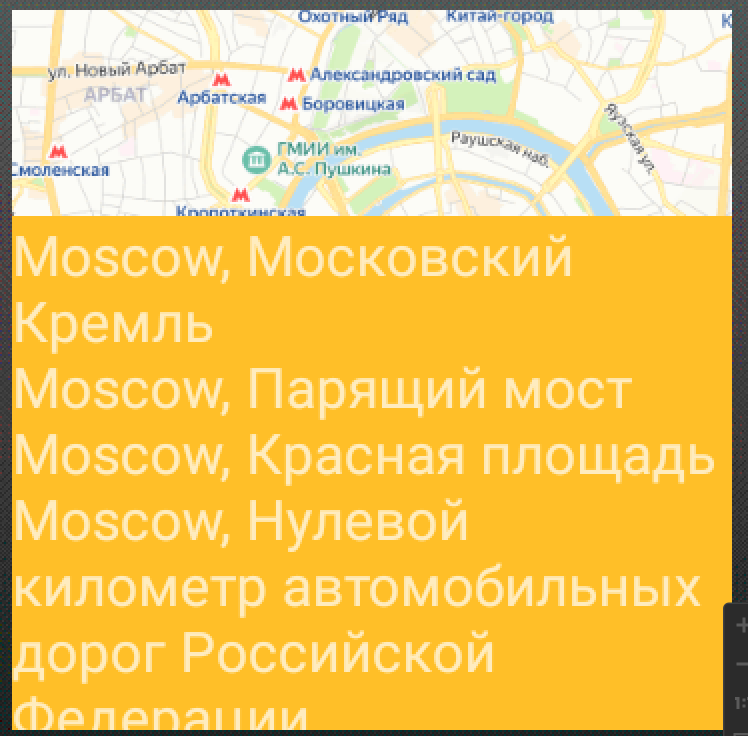
implementation 'androidx.lifecycle:lifecycle-livedata-ktx:2.5.1'  
  
implementation 'androidx.room:room-ktx:2.4.3'  
kapt 'androidx.room:room-compiler:2.4.3'  
implementation 'com.android.volley:volley:1.2.1'  
  
implementation 'com.squareup.retrofit2:retrofit:2.9.0'  
implementation 'com.squareup.retrofit2:converter-gson:2.9.0'  
  
implementation 'com.squareup.retrofit2:retrofit:2.9.0'  
implementation 'com.squareup.retrofit2:converter-gson:2.9.0'  
implementation 'com.squareup.retrofit2:converter-scalars:2.9.0'  
  
implementation 'com.squareup.picasso:picasso:2.71828'  
implementation 'com.google.android.gms:play-services-maps:18.2.0'  
implementation 'com.google.maps.android:android-maps-utils:0.5'  
  
implementation 'androidx.core:core-ktx:1.7.0'  
implementation 'com.google.android.gms:play-services-wearable:18.1.0'  
implementation 'androidx.percentlayout:percentlayout:1.0.0'  
implementation 'androidx.legacy:legacy-support-v4:1.0.0'  
implementation 'androidx.recyclerview:recyclerview:1.3.2'  
implementation 'androidx.wear:wear:1.3.0'  
implementation 'androidx.appcompat:appcompat:1.6.1'  
implementation 'com.google.android.material:material:1.4.0'  
implementation 'androidx.constraintlayout:constraintlayout:2.1.4'

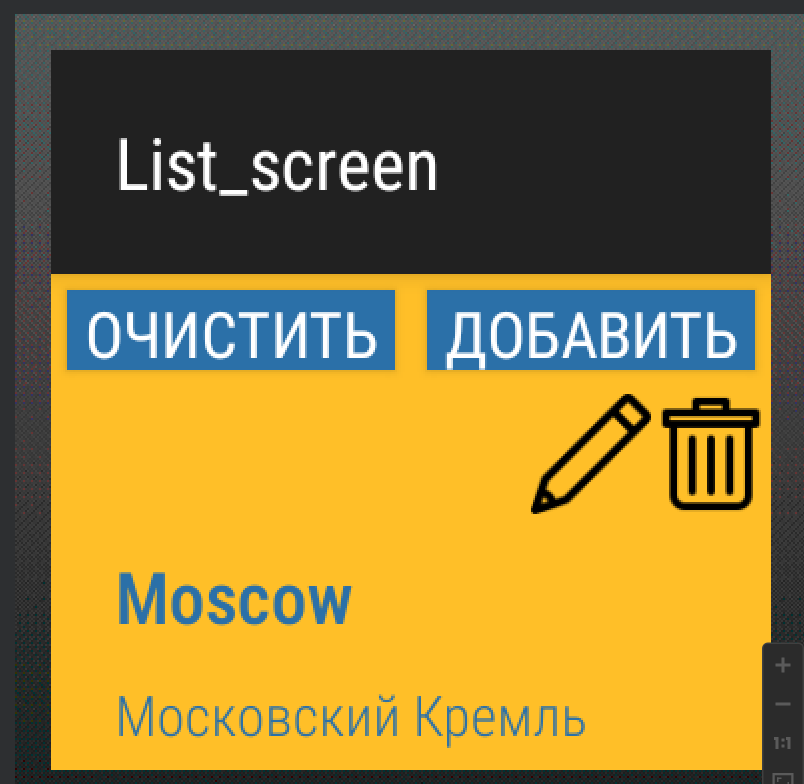
**Экраны**

****

****

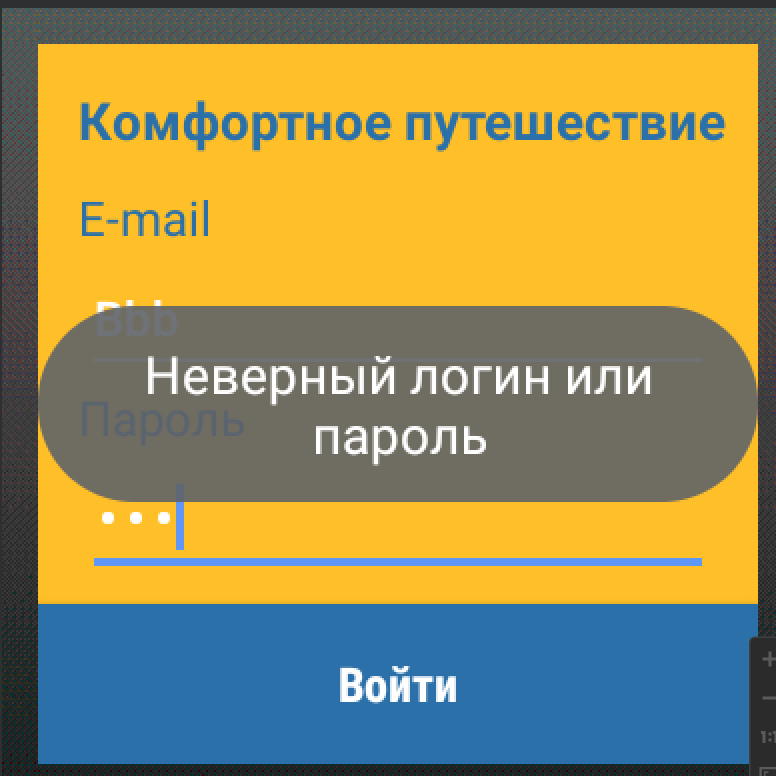
****

****

****

**Тестовые ситуации**

Проверка логина и пароля



## **Используемые инструменты**

kotlin, xml, android studio

## **Вывод:**

Я разработала мобильное приложение – умные часы, используя запросы api,базу данных room и sharedpreferens