Министерство образования и молодежной политики Свердловской области



ГАПОУ СО «Екатеринбургский колледж транспортного строительства»

Отчёт по УП

«Индивидуальные задания по теме : базы данных в Android Studio»

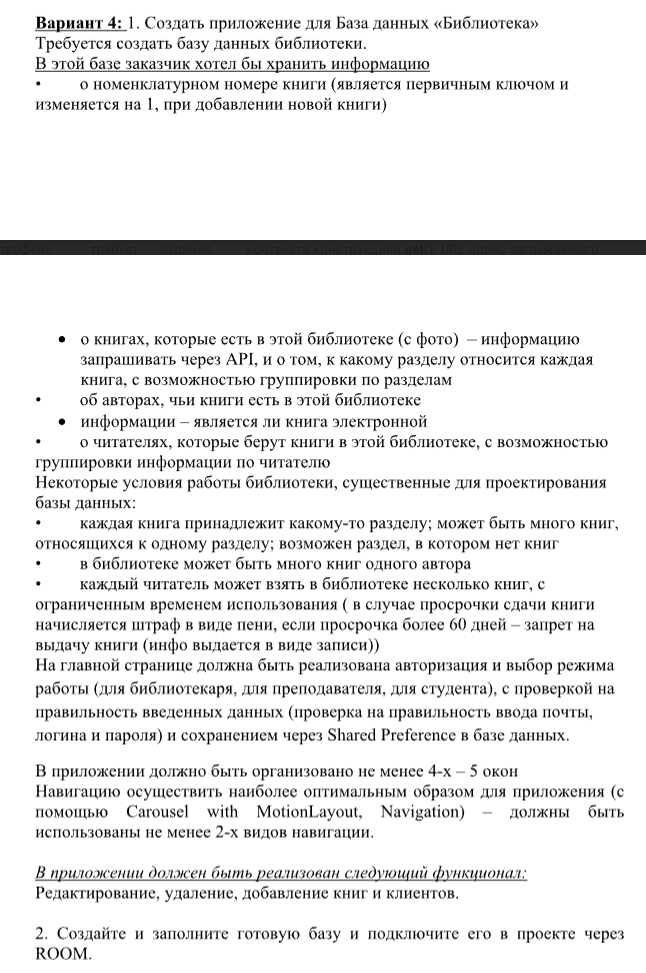
Выполнила: Сергеева Агата

Группа: ПР-31

Преподаватель: Мирошниченко Г.В

2024

**Описание задачи:**

****

**Входные данные:** email – почта пользователя, string; login – логин пользователя, string; password – пароль пользователя, string; name – имя пользователя, string; lastname – фамилия пользователя, string; newBook – название новой книги, string; elect – формат книги (электронная/ физическая), bool; recommend – статус книги ( рекомендованная / не рекомендованная), bool.

**Выходные данные:** img – обложка книги, image; title – название книги, string; author – автор книги, string.

**Структура проекта:**

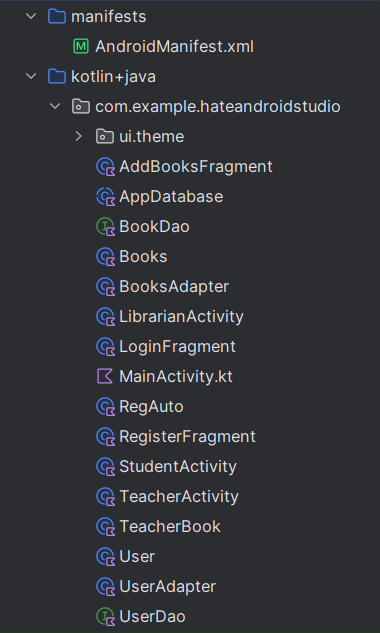
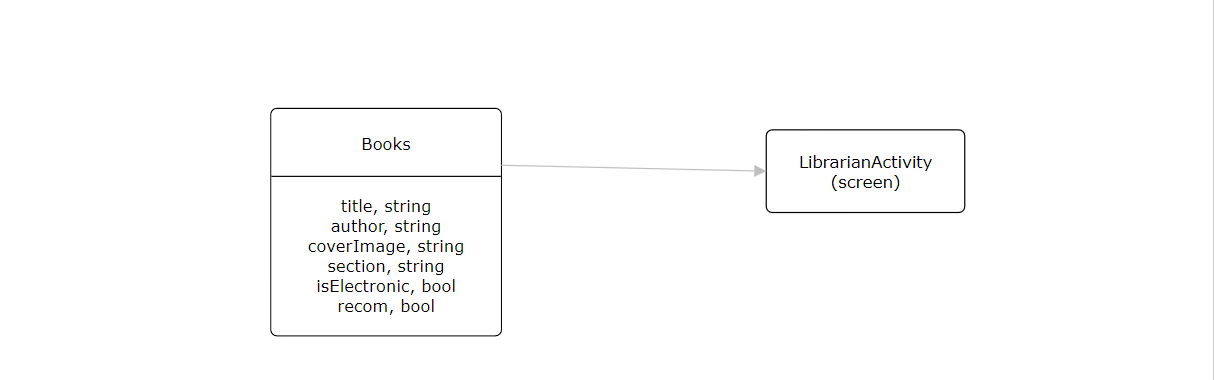


Схема классов

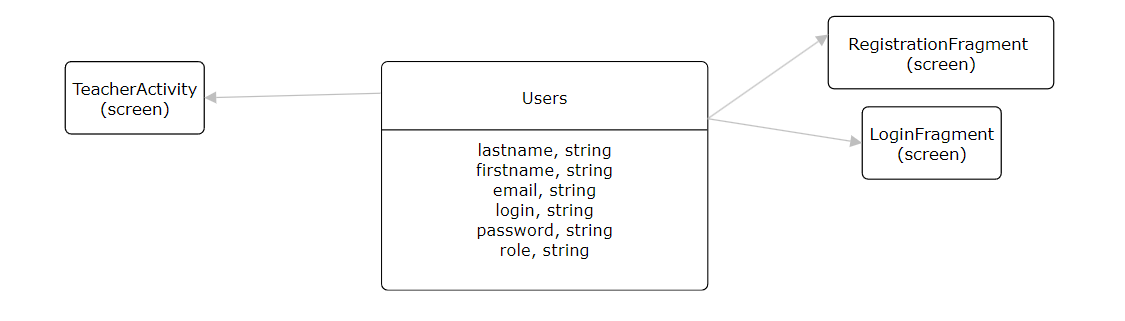


1

M

ER-диаграмма №1

1



1

1

M

M

M

ER-диаграмма №2

**Листинг программы:**

**XML-файлы:**

**activity\_librarian**

<?xml version="1.0" encoding="utf-8"?>  
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:background="@color/white"  
 android:id="@+id/fragmentContainer">  
</RelativeLayout>

**activity\_reg\_auto**

<?xml version="1.0" encoding="utf-8"?>  
<androidx.constraintlayout.widget.ConstraintLayout 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:background="@color/white"  
 tools:context=".RegAuto">  
   
 <FrameLayout  
 android:id="@+id/fragmentContainer"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent" />  
  
</androidx.constraintlayout.widget.ConstraintLayout>

**activity\_student**

<?xml version="1.0" encoding="utf-8"?>  
<androidx.constraintlayout.widget.ConstraintLayout 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:background="@color/white"  
 tools:context=".StudentActivity">  
  
 <ImageView  
 android:id="@+id/onLog"  
 android:layout\_width="50dp"  
 android:layout\_height="50dp"  
 android:src="@drawable/house"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintHorizontal\_bias="0.019"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 app:layout\_constraintVertical\_bias="0.01"  
 tools:ignore="MissingConstraints" />  
  
 <TextView  
 android:id="@+id/textView"  
 android:layout\_width="251dp"  
 android:layout\_height="57dp"  
 android:layout\_gravity="center"  
 android:paddingBottom="16dp"  
 android:text="Книги"  
 android:textAlignment="center"  
 android:textColor="@color/black"  
 android:textSize="35sp"  
 android:textStyle="bold"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 app:layout\_constraintVertical\_bias="0.023"  
 tools:ignore="MissingConstraints" />  
 <androidx.recyclerview.widget.RecyclerView  
 android:id="@+id/libView"  
 android:layout\_width="350dp"  
 android:layout\_height="232dp"  
 android:paddingLeft="30dp"  
 android:paddingTop="10dp"  
 android:paddingBottom="10dp"  
 android:nestedScrollingEnabled="false"  
 android:background="@drawable/recycle"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintHorizontal\_bias="0.491"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 app:layout\_constraintVertical\_bias="0.146"  
 tools:ignore="MissingConstraints" />  
  
 <Button  
 android:id="@+id/removeButton"  
 android:layout\_width="200dp"  
 android:layout\_height="wrap\_content"  
 android:background="@drawable/button"  
 android:text="Взять"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintHorizontal\_bias="0.497"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 app:layout\_constraintVertical\_bias="0.929"  
 tools:ignore="MissingConstraints" />  
  
 <TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_gravity="center"  
 android:paddingBottom="16dp"  
 android:text="Рекомендации"  
 android:textColor="@color/black"  
 android:textSize="35sp"  
 android:textStyle="bold"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintHorizontal\_bias="0.496"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 app:layout\_constraintVertical\_bias="0.483"  
 tools:ignore="MissingConstraints" />  
  
 <androidx.recyclerview.widget.RecyclerView  
 android:id="@+id/recView"  
 android:layout\_width="350dp"  
 android:layout\_height="232dp"  
 android:paddingLeft="30dp"  
 android:paddingTop="10dp"  
 android:paddingBottom="10dp"  
 android:nestedScrollingEnabled="false"  
 android:background="@drawable/recycle"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintHorizontal\_bias="0.491"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 app:layout\_constraintVertical\_bias="0.753"  
 tools:ignore="MissingConstraints" />  
  
</androidx.constraintlayout.widget.ConstraintLayout>

**activity\_teacher**

<?xml version="1.0" encoding="utf-8"?>  
<androidx.constraintlayout.widget.ConstraintLayout 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"  
 tools:context=".TeacherActivity">  
  
 <ImageView  
 android:id="@+id/onLog"  
 android:layout\_width="50dp"  
 android:layout\_height="50dp"  
 android:src="@drawable/house"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 tools:ignore="MissingConstraints" />  
 <LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical">  
 <TextView  
 android:layout\_width="251dp"  
 android:layout\_height="57dp"  
 android:layout\_gravity="center"  
 android:layout\_marginTop="10dp"  
 android:text="Новый студент"  
 android:textAlignment="center"  
 android:textColor="@color/black"  
 android:textSize="30sp"  
 android:textStyle="bold"  
 tools:ignore="MissingConstraints" />  
  
 <EditText  
 android:id="@+id/lastNameEditText"  
 android:layout\_marginTop="10dp"  
 android:layout\_marginLeft="30dp"  
 android:layout\_width="350dp"  
 android:layout\_height="wrap\_content"  
 android:hint="Фамилия" />  
  
 <EditText android:id="@+id/firstNameEditText"  
 android:layout\_marginLeft="30dp"  
 android:layout\_width="350dp"  
 android:layout\_height="wrap\_content"  
 android:hint="Имя"/>  
  
 <EditText android:id="@+id/emailEditText"  
 android:layout\_marginLeft="30dp"  
 android:layout\_width="350dp"  
 android:layout\_height="wrap\_content"  
 android:inputType="textEmailAddress"  
 android:hint="Почта"/>  
  
 <EditText android:id="@+id/loginEditText"  
 android:layout\_marginLeft="30dp"  
 android:layout\_width="350dp"  
 android:layout\_height="wrap\_content"  
 android:hint="Логин"/>  
  
 <EditText android:id="@+id/passwordEditText"  
 android:layout\_marginLeft="30dp"  
 android:layout\_width="350dp"  
 android:layout\_height="wrap\_content"  
 android:hint="Пароль" android:inputType="textPassword"/>  
  
 <Button  
 android:id="@+id/registerButton"  
 android:layout\_width="250dp"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginLeft="80dp"  
 android:layout\_marginTop="10dp"  
 android:background="@drawable/button"  
 android:onClick="SaveMeGod"  
 android:text="Зарегистрировать студента" />  
  
 <androidx.recyclerview.widget.RecyclerView  
 android:id="@+id/recyclerView"  
 android:layout\_width="350dp"  
 android:layout\_height="252dp"  
 android:layout\_marginTop="10dp"  
 android:layout\_marginLeft="30dp"  
 android:layout\_marginBottom="10dp"  
 android:background="@drawable/recycle"  
 android:nestedScrollingEnabled="false"  
 android:paddingLeft="30dp"  
 android:paddingTop="10dp"  
 tools:ignore="MissingConstraints" />  
  
 <Button  
 android:id="@+id/lib"  
 android:layout\_width="120dp"  
 android:layout\_height="wrap\_content"  
 android:text="Библиотека"  
 android:layout\_marginTop="30dp"  
 android:layout\_marginLeft="150dp"  
 android:background="@drawable/button"  
 tools:ignore="MissingConstraints" />  
  
  
 </LinearLayout>  
  
</androidx.constraintlayout.widget.ConstraintLayout>

**activity\_teacher\_book**

<?xml version="1.0" encoding="utf-8"?>  
<androidx.constraintlayout.widget.ConstraintLayout 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"  
 tools:context=".TeacherBook">  
  
 <ImageView  
 android:id="@+id/onLog"  
 android:layout\_width="50dp"  
 android:layout\_height="50dp"  
 android:src="@drawable/streloc"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintHorizontal\_bias="0.019"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 app:layout\_constraintVertical\_bias="0.01"  
 tools:ignore="MissingConstraints" />  
  
 <TextView  
 android:id="@+id/textView"  
 android:layout\_width="251dp"  
 android:layout\_height="57dp"  
 android:layout\_gravity="center"  
 android:paddingBottom="16dp"  
 android:text="Книги"  
 android:textAlignment="center"  
 android:textColor="@color/black"  
 android:textSize="35sp"  
 android:textStyle="bold"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 app:layout\_constraintVertical\_bias="0.023"  
 tools:ignore="MissingConstraints" />  
  
 <androidx.recyclerview.widget.RecyclerView  
 android:id="@+id/libView"  
 android:layout\_width="350dp"  
 android:layout\_height="232dp"  
 android:background="@drawable/recycle"  
 android:nestedScrollingEnabled="false"  
 android:paddingLeft="30dp"  
 android:paddingTop="10dp"  
 android:paddingBottom="10dp"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintHorizontal\_bias="0.655"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 app:layout\_constraintVertical\_bias="0.146"  
 tools:ignore="MissingConstraints" />  
  
 <TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_gravity="center"  
 android:paddingBottom="16dp"  
 android:text="Рекомендации"  
 android:textColor="@color/black"  
 android:textSize="35sp"  
 android:textStyle="bold"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintHorizontal\_bias="0.496"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 app:layout\_constraintVertical\_bias="0.483"  
 tools:ignore="MissingConstraints" />  
  
 <androidx.recyclerview.widget.RecyclerView  
 android:id="@+id/recView"  
 android:layout\_width="350dp"  
 android:layout\_height="232dp"  
 android:background="@drawable/recycle"  
 android:nestedScrollingEnabled="false"  
 android:paddingLeft="30dp"  
 android:paddingTop="10dp"  
 android:paddingBottom="10dp"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintHorizontal\_bias="0.655"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 app:layout\_constraintVertical\_bias="0.773"  
 tools:ignore="MissingConstraints" />  
   
</androidx.constraintlayout.widget.ConstraintLayout>

**fragment\_add\_books**

<?xml version="1.0" encoding="utf-8"?>  
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 tools:context=".AddBooksFragment">  
  
 <ImageView  
 android:id="@+id/onLog"  
 android:layout\_width="40dp"  
 android:layout\_height="40dp"  
 android:layout\_marginTop="20dp"  
 android:layout\_marginLeft="5dp"  
 android:src="@drawable/house"/>  
  
 <LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical">  
  
 <LinearLayout  
 android:layout\_width="350dp"  
 android:layout\_height="wrap\_content"  
 android:layout\_gravity="center"  
 android:layout\_marginLeft="20dp"  
 android:layout\_marginTop="20dp"  
 android:orientation="horizontal">  
  
 <EditText  
 android:id="@+id/titleEditText"  
 android:layout\_width="0dp"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginEnd="8dp"  
 android:layout\_weight="1"  
 android:hint="Название книги" /> <!-- Отступ между EditText и Button -->  
  
 <Button  
 android:id="@+id/searchButton"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:background="@drawable/button"  
 android:text="Найти" />  
 </LinearLayout>  
  
 <ImageView  
 android:id="@+id/coverImageView"  
 android:layout\_width="350dp"  
 android:layout\_height="100dp"  
 android:layout\_marginLeft="35dp" />  
  
 <TextView  
 android:id="@+id/titleTextView"  
 android:layout\_width="350dp"  
 android:layout\_height="30dp"  
 android:layout\_marginLeft="35dp"  
 android:textAlignment="center"  
 android:textColor="@color/black"  
 android:textSize="15dp"  
 android:textStyle="bold" />  
  
 <TextView  
 android:id="@+id/authorTextView"  
 android:layout\_width="350dp"  
 android:layout\_height="30dp"  
 android:layout\_marginLeft="35dp"  
 android:textAlignment="center"  
 android:textColor="@color/black"  
 android:textSize="15dp" />  
  
 <EditText  
 android:id="@+id/sectionEditText"  
 android:layout\_width="350dp"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginLeft="35dp"  
 android:hint="Раздел"  
 android:textColor="@color/black" />  
  
 <EditText  
 android:id="@+id/daysEditText"  
 android:layout\_width="350dp"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginLeft="35dp"  
 android:hint="Число дней"  
 android:inputType="number"  
 android:textColor="@color/black" />  
  
 <CheckBox  
 android:id="@+id/electronicCheckBox"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginLeft="35dp"  
 android:text="Электронная" />  
  
 <CheckBox  
 android:id="@+id/recomendationCheckBox"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginLeft="35dp"  
 android:text="Рекомендовать" />  
  
 <Button  
 android:id="@+id/saveButton"  
 android:layout\_width="100dp"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginLeft="150dp"  
 android:layout\_marginTop="10dp"  
 android:background="@drawable/button"  
 android:text="Сохранить" />  
  
  
 </LinearLayout>  
  
 <androidx.recyclerview.widget.RecyclerView  
 android:id="@+id/recyclerView"  
 android:layout\_width="350dp"  
 android:layout\_height="200dp"  
 android:background="@drawable/recycle"  
 android:paddingLeft="30dp"  
 android:paddingTop="10dp"  
 android:paddingBottom="10dp"  
 android:layout\_marginTop="450dp"  
 android:layout\_marginLeft="35dp"  
 android:nestedScrollingEnabled="false" />  
 <Button  
 android:id="@+id/deleteButton"  
 android:layout\_width="200dp"  
 android:layout\_marginTop="660dp"  
 android:layout\_marginLeft="35dp"  
 android:layout\_height="wrap\_content"  
 android:background="@drawable/button"  
 android:text="Удалить выбранные" />  
  
 <Button  
 android:id="@+id/clearButton"  
 android:layout\_width="100dp"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginLeft="285dp"  
 android:layout\_marginTop="660dp"  
 android:background="@drawable/button"  
 android:text="Очистить базу" />  
</FrameLayout>

**fragment\_login**

<?xml version="1.0" encoding="utf-8"?>  
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:background="@color/white"  
 tools:context=".LoginFragment">  
  
 <LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical"  
 android:padding="16dp">  
  
 <TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Авторизация"  
 android:textSize="24sp"  
 android:textStyle="bold"  
 android:textColor="@color/black"  
 android:layout\_gravity="center"  
 android:paddingBottom="16dp"/>  
  
 <EditText  
 android:id="@+id/emailEditText"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="10dp"  
 android:hint="Почта"  
 android:inputType="textEmailAddress"/>  
  
 <EditText  
 android:id="@+id/loginEditText"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:hint="Логин"  
 android:inputType="text"/>  
  
 <EditText  
 android:id="@+id/passwordEditText"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:hint="Пароль"  
 android:inputType="textPassword"/>  
  
 <Spinner  
 android:id="@+id/roleSpinner"  
 android:layout\_width="match\_parent"  
 android:layout\_height="45dp"  
 android:background="@drawable/button"/>  
  
 <Button  
 android:id="@+id/loginButton"  
 android:layout\_width="250dp"  
 android:layout\_height="wrap\_content"  
 android:background="@drawable/button"  
 android:text="Войти"  
 android:layout\_marginTop="16dp"  
 android:layout\_marginLeft="60dp"/>  
 <LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="40dp"  
 android:orientation="horizontal"  
 android:layout\_gravity="center"  
 android:layout\_marginLeft="52dp"  
 android:gravity="center\_vertical">  
  
 <TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="35dp"  
 android:layout\_marginTop="8dp"  
 android:text="Нет аккаунта?"  
 android:textSize="14sp"  
 android:textColor="@color/black"  
 android:paddingBottom="16dp" />  
  
 <Button  
 android:id="@+id/registerButton"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginLeft="8dp"  
 android:textSize="10sp"  
 android:textStyle="bold"  
 android:background="@drawable/butt"  
 android:text="Зарегистрироваться" />  
 </LinearLayout>  
  
 </LinearLayout>  
 <ImageView  
 android:id="@+id/coverImageView"  
 android:layout\_width="350dp"  
 android:layout\_height="400dp"  
 android:layout\_gravity="bottom"  
 android:layout\_marginLeft="35dp"  
 android:src="@drawable/base"/>  
  
</FrameLayout>

**fragment\_register**

<?xml version="1.0" encoding="utf-8"?>  
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:background="@color/white"  
 tools:context=".RegisterFragment">  
 <ImageView  
 android:id="@+id/onLog"  
 android:layout\_width="50dp"  
 android:layout\_height="50dp"  
 android:layout\_marginTop="5dp"  
 android:layout\_marginLeft="5dp"  
 android:src="@drawable/streloc"/>  
  
 <LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical"  
 android:padding="16dp">  
 <TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Регистрация"  
 android:textSize="24sp"  
 android:textStyle="bold"  
 android:textColor="@color/black"  
 android:layout\_gravity="center"  
 android:paddingBottom="16dp"/>  
 <EditText  
 android:id="@+id/lastNameEditText"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="10dp"  
 android:hint="Фамилия" />  
  
 <EditText  
 android:id="@+id/firstNameEditText"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:hint="Имя" />  
  
 <EditText  
 android:id="@+id/emailEditText"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:hint="Почта"  
 android:inputType="textEmailAddress" />  
  
 <EditText  
 android:id="@+id/loginEditText"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:hint="Логин" />  
  
 <EditText  
 android:id="@+id/passwordEditText"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:hint="Пароль"  
 android:inputType="textPassword" />  
  
 <Spinner  
 android:id="@+id/roleSpinner"  
 android:layout\_width="match\_parent"  
 android:layout\_height="45dp"  
 android:background="@drawable/button" />  
  
 <Button  
 android:id="@+id/registerButton"  
 android:layout\_width="200dp"  
 android:layout\_height="wrap\_content"  
 android:background="@drawable/button"  
 android:text="Зарегистрироваться"  
 android:layout\_marginTop="16dp"  
 android:layout\_marginLeft="85dp"/>  
 </LinearLayout>  
  
</FrameLayout>

**item\_book**

<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:layout\_width="310dp"  
 android:layout\_height="135dp"  
 android:orientation="vertical"  
 android:background="#FFFFFF"  
 android:padding="2dp">  
  
 <TextView  
 android:id="@+id/bookTitleTextView"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:background="@android:color/transparent"  
 android:textStyle="bold"/>  
  
 <TextView  
 android:id="@+id/bookAuthorTextView"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:background="@android:color/transparent"/>  
  
 <TextView  
 android:id="@+id/r"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:background="@android:color/transparent"/>  
  
 <TextView  
 android:id="@+id/day"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:background="@android:color/transparent"/>  
  
 <TextView  
 android:id="@+id/bookCheckBox"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:background="@android:color/transparent"/>  
  
 <TextView  
 android:id="@+id/recom"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:background="@android:color/transparent"/>  
  
 <TextView  
 android:id="@+id/zzz"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:background="@android:color/transparent"  
 android:text="-----------------------------------------------------"  
 android:textStyle="bold"/>  
</LinearLayout>

**item\_student**

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:layout\_width="310dp"  
 android:layout\_height="135dp"  
 android:orientation="vertical"  
 android:background="#FFFFFF"  
 android:padding="16dp">  
  
 <TextView  
 android:id="@+id/fam"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:background="@android:color/transparent"  
 android:textSize="16sp" />  
  
 <TextView  
 android:id="@+id/name"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:background="@android:color/transparent"  
 android:textSize="16sp" />  
  
 <TextView  
 android:id="@+id/mail"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:background="@android:color/transparent"  
 android:textSize="16sp" />  
</LinearLayout>

**KT-файлы:**

**AddBooksFragment**

class AddBooksFragment : Fragment() {  
 private var param1: String? = null  
 private var param2: String? = null  
  
 private lateinit var booksAdapter: BooksAdapter  
 private lateinit var recyclerView: RecyclerView  
 private val booksList = *mutableListOf*<Books>()  
 private lateinit var titleTextView: TextView  
 private lateinit var titleEditText: EditText  
 private lateinit var sectionEditText: EditText  
 private lateinit var daysEditText: EditText  
 private lateinit var electronicCheckBox: CheckBox  
 private lateinit var coverImageView: ImageView  
 private lateinit var authorTextView: TextView  
 private lateinit var saveButton: Button  
 private lateinit var searchButton: Button  
 private lateinit var removeButton: Button  
 private lateinit var clearButton: Button  
 private lateinit var database: AppDatabase  
 private lateinit var rec: CheckBox  
  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 *arguments*?.*let* **{** param1 = **it**.getString(*ARG\_PARAM1*)  
 param2 = **it**.getString(*ARG\_PARAM2*)  
 **}** }  
  
 @SuppressLint("MissingInflatedId")  
 override fun onCreateView(  
 inflater: LayoutInflater, container: ViewGroup?,  
 savedInstanceState: Bundle?  
 ): View {  
 val view = inflater.inflate(R.layout.*fragment\_add\_books*, container, false)  
  
 removeButton = view.findViewById(R.id.*deleteButton*)  
 titleTextView = view.findViewById(R.id.*titleTextView*)  
 sectionEditText = view.findViewById(R.id.*sectionEditText*)  
 titleEditText = view.findViewById(R.id.*titleEditText*)  
 daysEditText = view.findViewById(R.id.*daysEditText*)  
 electronicCheckBox = view.findViewById(R.id.*electronicCheckBox*)  
 coverImageView = view.findViewById(R.id.*coverImageView*)  
 authorTextView = view.findViewById(R.id.*authorTextView*)  
 saveButton = view.findViewById(R.id.*saveButton*)  
 searchButton = view.findViewById(R.id.*searchButton*)  
 clearButton = view.findViewById(R.id.*clearButton*)  
 rec = view.findViewById(R.id.*recomendationCheckBox*)  
  
 database = Room.databaseBuilder(requireContext(), AppDatabase::class.*java*, "books.db").build()  
 val bookDao = database.bookDao()  
  
 loadBooks()  
  
 saveButton.setOnClickListener **{** saveBook() **}** searchButton.setOnClickListener **{** fetchBookInfo() **}** clearButton.setOnClickListener **{** clearBooks() **}** recyclerView = view.findViewById(R.id.*recyclerView*)  
 booksAdapter = BooksAdapter(booksList, **{** book **-> }**, **{** book **-> }**)  
 recyclerView.*adapter* = booksAdapter  
 recyclerView.*layoutManager* = LinearLayoutManager(requireContext())  
  
 removeButton.setOnClickListener **{** booksAdapter.removeSelectedBook(bookDao, *viewLifecycleOwner*.*lifecycleScope*)  
 **}** view.findViewById<ImageView>(R.id.*onLog*).setOnClickListener **{** val intent = Intent(requireActivity(), RegAuto::class.*java*)  
 startActivity(intent)  
 **}** return view  
 }  
  
 private fun loadBooks() {  
 *lifecycleScope*.*launch* **{** val bookDao = database.bookDao()  
 val savedBooks = bookDao.getAllBooks()  
 booksList.clear()  
 booksList.addAll(savedBooks)  
 booksAdapter.notifyDataSetChanged()  
 **}** }  
  
 private fun saveBook() {  
 val title = titleTextView.*text*.toString()  
 val author = authorTextView.*text*.toString().*removePrefix*("Director: ")  
 val section = sectionEditText.*text*.toString()  
 val maxDays = daysEditText.*text*.toString().*toIntOrNull*() ?: 0  
 val isElectronic = electronicCheckBox.*isChecked* val rec = rec.*isChecked* if (title.*isEmpty*() || author.*isEmpty*() || section.*isEmpty*() || maxDays.toString().*isEmpty*()) {  
 Toast.makeText(requireContext(), "Пожалуйста, заполните все поля", Toast.*LENGTH\_SHORT*).show()  
 }  
 else  
 {  
 if (maxDays < 1 || maxDays > 30)  
 {  
 Toast.makeText(requireContext(), "Количество дней некоррктно", Toast.*LENGTH\_SHORT*).show()  
 }  
 else {  
 val book = Books(  
 title = title,  
 author = author,  
 coverImage = "",  
 section = section,  
 maxDays = maxDays,  
 isElectronic = isElectronic,  
 recom = rec  
 )  
  
 *lifecycleScope*.*launch* **{** val bookDao = database.bookDao()  
 bookDao.insert(book)  
  
  
 booksAdapter.addBook(book)  
 Toast.makeText(  
 requireContext(),  
 "Книга добавлена в библиотеку!",  
 Toast.*LENGTH\_SHORT* ).show()  
 **}** }  
 }  
 }  
  
 private fun fetchBookInfo() {  
 if (titleEditText.*text*.toString().*isNotEmpty*()) {  
 val title = titleEditText.*text*.toString()  
 val url = "https://www.omdbapi.com/?apikey=8424b5c9&t=${title.*replace*(" ", "+")}"  
  
 val queue = Volley.newRequestQueue(requireContext())  
 val stringRequest = StringRequest(  
 Request.Method.*GET*, url,  
 **{** response **->** val obj = JSONObject(response)  
 if (obj.getString("Response") == "True") {  
 val movieTitle = obj.getString("Title")  
 val authors = obj.getString("Director")  
 val coverUrl = obj.optString("Poster", null)  
  
 titleTextView.*text* = movieTitle  
 authorTextView.*text* = "Director: $authors"  
  
 if (coverUrl.*isNotEmpty*()) {  
 Glide.with(requireContext())  
 .load(coverUrl)  
 .into(coverImageView)  
 } else {  
 coverImageView.setImageResource(R.drawable.*zxc*)  
 }  
 } else {  
 titleTextView.*text* = "Книга не найдена"  
 authorTextView.*text* = ""  
 coverImageView.setImageResource(R.drawable.*zxc*)  
 }  
 **}**,  
 **{** error **->** titleTextView.*text* = "Какие-то проблемы?"  
 authorTextView.*text* = ""  
 coverImageView.setImageResource(R.drawable.*zxc*)  
 **}** )  
  
 queue.add(stringRequest)  
 } else {  
 Toast.makeText(  
 requireContext(),  
 "Пожалуйста, введите название книги!",  
 Toast.*LENGTH\_LONG* ).show()  
 }  
 }  
  
 private fun clearBooks() {  
 *lifecycleScope*.*launch* **{** val bookDao = database.bookDao()  
 bookDao.deleteAllBooks()  
 booksList.clear()  
 booksAdapter.notifyDataSetChanged()  
 Toast.makeText(requireContext(), "Все книги удалены!", Toast.*LENGTH\_SHORT*).show()  
 **}** }

}

**AppDatabase**

@Database(entities = [User::class, Books::class], version = 1)  
abstract class AppDatabase : RoomDatabase() {  
  
 abstract fun userDao(): UserDao  
  
 abstract fun bookDao(): BookDao  
}

**BookDao**

@Dao  
interface BookDao {  
 @Update  
 suspend fun updateRecommendation(book: Books)  
  
 @Delete  
 suspend fun delete(book: Books)  
  
 @Insert  
 suspend fun insert(book: Books)  
  
 @Query("SELECT \* FROM books")  
 suspend fun getAllBooks(): List<Books>  
  
 @Query("SELECT \* FROM books WHERE id = :bookId")  
 suspend fun getBookById(bookId: Int): Books?  
  
 @Query("DELETE FROM books")  
 suspend fun deleteAllBooks()  
  
 @Query("UPDATE books SET recom = true WHERE id = :bookId")  
 suspend fun updateRecommendation(bookId: Int)  
}

**Books**

@Entity(tableName = "books")  
data class Books(  
 @PrimaryKey(autoGenerate = true) val id: Int = 0,  
 val title: String,  
 val author: String,  
 val coverImage: String,  
 val section: String,  
 val maxDays: Int,  
 val isElectronic: Boolean,  
 val recom: Boolean  
)

**BooksAdapter**

class BooksAdapter(  
 private val booksList: MutableList<Books>,  
 private val onDelete: (Books) -> Unit,  
 private val onItemClick: (Books) -> Unit  
) : RecyclerView.Adapter<BooksAdapter.BookViewHolder>() {  
  
 private var selectedPosition: Int = RecyclerView.*NO\_POSITION* inner class BookViewHolder(itemView: View) : RecyclerView.ViewHolder(itemView) {  
 val titleTextView: TextView = itemView.findViewById(R.id.*bookTitleTextView*)  
 val authorTextView: TextView = itemView.findViewById(R.id.*bookAuthorTextView*)  
 val r: TextView = itemView.findViewById(R.id.*r*)  
 val day: TextView = itemView.findViewById(R.id.*day*)  
 var checkBox: TextView = itemView.findViewById(R.id.*bookCheckBox*)  
 var recom: TextView = itemView.findViewById(R.id.*recom*)  
  
 init {  
 itemView.setOnClickListener **{** selectedPosition = if (*adapterPosition* == selectedPosition) {  
 RecyclerView.*NO\_POSITION* } else {  
 *adapterPosition* }  
 notifyDataSetChanged()  
 **}** }  
  
 fun bind(book: Books) {  
 titleTextView.*text* = book.title  
 authorTextView.*text* = book.author  
 r.*text* = book.section  
 day.*text* = book.maxDays.toString()  
 checkBox.*text* = if (book.isElectronic != true) "Физическая" else "Электронная"  
 recom.*text* = book.recom.toString()  
 }  
 }  
  
 override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): BookViewHolder {  
 val view = LayoutInflater.from(parent.*context*).inflate(R.layout.*item\_book*, parent, false)  
 return BookViewHolder(view)  
 }  
  
 override fun onBindViewHolder(holder: BookViewHolder, position: Int) {  
 holder.bind(booksList[position])  
 holder.itemView.setBackgroundColor(  
 if (position == selectedPosition) Color.*MAGENTA* else Color.*WHITE* )  
 }  
  
 override fun getItemCount(): Int {  
 return booksList.size  
 }  
  
 fun addBook(book: Books) {  
 booksList.add(book)  
 notifyItemInserted(booksList.size - 1)  
 }  
  
 fun removeSelectedBook(bookDao: BookDao, lifecycleScope: CoroutineScope) {  
 if (selectedPosition != RecyclerView.*NO\_POSITION*) {  
 val bookToRemove = booksList[selectedPosition]  
 booksList.removeAt(selectedPosition)  
 notifyItemRemoved(selectedPosition)  
 selectedPosition = RecyclerView.*NO\_POSITION* onDelete(bookToRemove)  
  
 lifecycleScope.*launch* **{** bookDao.delete(bookToRemove)  
 **}** }  
 }  
}

**LibrarianActivity**

class LibrarianActivity : AppCompatActivity() {  
  
 @SuppressLint("MissingInflatedId")  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_librarian*)  
  
 if (savedInstanceState == null) {  
 *supportFragmentManager*.beginTransaction()  
 .replace(R.id.*fragmentContainer*, AddBooksFragment())  
 .commit()  
 }  
 }  
}

**LoginFragment**

class LoginFragment : Fragment() {  
 // *TODO: Rename and change types of parameters* private var param1: String? = null  
 private var param2: String? = null  
  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 *arguments*?.*let* **{** param1 = **it**.getString(*ARG\_PARAM1*)  
 param2 = **it**.getString(*ARG\_PARAM2*)  
 **}** }  
  
 private lateinit var db: AppDatabase  
  
 @SuppressLint("MissingInflatedId")  
 override fun onCreateView(  
 inflater: LayoutInflater, container: ViewGroup?,  
 savedInstanceState: Bundle?  
 ): View? {  
 val view = inflater.inflate(R.layout.*fragment\_login*, container, false)  
 db = Room.databaseBuilder(requireContext(), AppDatabase::class.*java*, "user-db").build()  
  
 val roleSpinner: Spinner = view.findViewById(R.id.*roleSpinner*)  
 val roles = *arrayOf*("Библиотекарь", "Преподаватель", "Студент")  
 val adapter = ArrayAdapter(requireContext(), android.R.layout.*simple\_spinner\_item*, roles)  
 adapter.setDropDownViewResource(android.R.layout.*simple\_spinner\_dropdown\_item*)  
 roleSpinner.*adapter* = adapter  
  
 view.findViewById<Button>(R.id.*loginButton*).setOnClickListener **{** val email = view.findViewById<EditText>(R.id.*emailEditText*).*text*.toString()  
 val login = view.findViewById<EditText>(R.id.*loginEditText*).*text*.toString()  
 val password = view.findViewById<EditText>(R.id.*passwordEditText*).*text*.toString()  
 val role = roleSpinner.*selectedItem*.toString()  
  
  
 if (email.*isNotEmpty*() && login.*isNotEmpty*() && password.*isNotEmpty*()) {  
 *lifecycleScope*.*launch* **{** val userCount = db.userDao().getUserCount()  
 if (userCount == 0) {  
 Toast.makeText(requireContext(), "База данных пуста. Пожалуйста, зарегистрируйтесь.", Toast.*LENGTH\_SHORT*).show()  
 return@launch  
 }  
 val emailPattern = Regex("^[A-Za-z0-9.\_%+-]+@[A-Za-z0-9.-]+\\.[A-Z|a-z]{2,}$")  
 if(!emailPattern.matches(email)){  
 Toast.makeText(requireContext(), "Введите корректный адрес электронной почты", Toast.*LENGTH\_SHORT*).show()  
 }  
 else {  
 val user = db.userDao().getUser(email, login, password, role)  
 if (user != null) {  
 navigateToRoleScreen(user.role)  
 } else {  
 Toast.makeText(requireContext(), "Неверные данные", Toast.*LENGTH\_SHORT*)  
 .show()  
 }  
 }  
 **}** } else {  
 Toast.makeText(requireContext(), "Пожалуйста, заполните все поля", Toast.*LENGTH\_SHORT*).show()  
 }  
 **}** view.findViewById<Button>(R.id.*registerButton*).setOnClickListener **{** val registerFragment = RegisterFragment()  
 *parentFragmentManager*.beginTransaction()  
 .replace(R.id.*fragmentContainer*, registerFragment)  
 .addToBackStack(null)  
 .commit()  
 **}** return view  
 }  
  
 private fun navigateToRoleScreen(role: String) {  
 when (role) {  
 "Библиотекарь" -> navigateToLibrarianScreen()  
 "Преподаватель" -> navigateToTeacherScreen()  
 "Студент" -> navigateToStudentScreen()  
 else -> Toast.makeText(requireContext(), "Неизвестная роль", Toast.*LENGTH\_SHORT*).show()  
 }  
 }  
  
 private fun navigateToLibrarianScreen() {  
  
 val intent = Intent(requireContext(), LibrarianActivity::class.*java*)  
 startActivity(intent)  
 }  
  
 private fun navigateToTeacherScreen() {  
  
 val intent = Intent(requireContext(), TeacherActivity::class.*java*)  
 startActivity(intent)  
 }  
  
 private fun navigateToStudentScreen() {  
 val intent = Intent(requireContext(), StudentActivity::class.*java*)  
 startActivity(intent)  
 }

}

**RegAuto**

class RegAuto : AppCompatActivity() {  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_reg\_auto*)  
  
 if (savedInstanceState == null) {  
 *supportFragmentManager*.beginTransaction()  
 .replace(R.id.*fragmentContainer*, LoginFragment())  
 .commit()  
 }  
 }  
}

**RegisterFrgment**

class RegisterFragment : Fragment() {  
 private var param1: String? = null  
 private var param2: String? = null  
  
 private lateinit var db: AppDatabase  
  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 *arguments*?.*let* **{** param1 = **it**.getString(*ARG\_PARAM1*)  
 param2 = **it**.getString(*ARG\_PARAM2*)  
 **}** }  
  
 @SuppressLint("MissingInflatedId")  
 override fun onCreateView(  
 inflater: LayoutInflater, container: ViewGroup?,  
 savedInstanceState: Bundle?  
 ): View? {  
 val view = inflater.inflate(R.layout.*fragment\_register*, container, false)  
  
 db = Room.databaseBuilder(requireContext(), AppDatabase::class.*java*, "user-db").build()  
  
 val roleSpinner: Spinner = view.findViewById(R.id.*roleSpinner*)  
 val roles = *arrayOf*("Библиотекарь", "Преподаватель", "Студент")  
 roleSpinner.*adapter* =  
 ArrayAdapter(requireContext(), android.R.layout.*simple\_spinner\_item*, roles)  
  
 view.findViewById<Button>(R.id.*registerButton*).setOnClickListener **{** val lastName = view.findViewById<EditText>(R.id.*lastNameEditText*).*text*.toString()  
 val firstName = view.findViewById<EditText>(R.id.*firstNameEditText*).*text*.toString()  
 val email = view.findViewById<EditText>(R.id.*emailEditText*).*text*.toString()  
 val login = view.findViewById<EditText>(R.id.*loginEditText*).*text*.toString()  
 val password = view.findViewById<EditText>(R.id.*passwordEditText*).*text*.toString()  
 val role = roleSpinner.*selectedItem*.toString()  
  
 if (lastName.*isEmpty*() || firstName.*isEmpty*() || email.*isEmpty*() || login.*isEmpty*() || password.*isEmpty*()) {  
 Toast.makeText(  
 requireContext(),  
 "Пожалуйста, заполните все поля",  
 Toast.*LENGTH\_SHORT* ).show()  
 } else {  
 val emailPattern = Regex("^[A-Za-z0-9.\_%+-]+@[A-Za-z0-9.-]+\\.[A-Z|a-z]{2,}$")  
 if (!emailPattern.matches(email)) {  
 Toast.makeText(  
 requireContext(),  
 "Введите корректный адрес электронной почты",  
 Toast.*LENGTH\_SHORT* ).show()  
 } else if (password.length < 8) {  
 Toast.makeText(  
 requireContext(),  
 "Пароль должен содержать не менее 8 символов",  
 Toast.*LENGTH\_SHORT* ).show()  
 } else {  
 *lifecycleScope*.*launch* **{** val studentCount = db.userDao()  
 .getStudentCount()  
 if (studentCount >= 25) {  
 Toast.makeText(  
 requireContext(),  
 "Достигнуто максимальное количество студентов (25)",  
 Toast.*LENGTH\_SHORT* ).show()  
 } else {  
 *lifecycleScope*.*launch* **{** val existingUser = db.userDao().findByLoginOrEmail(login, email)  
 if (existingUser != null) {  
 Toast.makeText(  
 requireContext(),  
 "Пользователь с таким логином или почтой уже существует",  
 Toast.*LENGTH\_SHORT* ).show()  
 } else if (role == "Преподаватель") {  
 val teacherCount =  
 db.userDao().countUsersByRole("Преподаватель")  
 if (teacherCount > 0) {  
 Toast.makeText(  
 requireContext(),  
 "Уже зарегистрирован один преподаватель",  
 Toast.*LENGTH\_SHORT* ).show()  
 } else {  
 val user = User(  
 lastName = lastName,  
 firstName = firstName,  
 email = email,  
 login = login,  
 password = password,  
 role = role  
 )  
 db.userDao().insert(user)  
 Toast.makeText(  
 requireContext(),  
 "Регистрация успешна",  
 Toast.*LENGTH\_SHORT* ).show()  
 }  
 } else if (role == "Библиотекарь") {  
 val librarianCount =  
 db.userDao().countUsersByRole("Библиотекарь")  
 if (librarianCount > 0) {  
 Toast.makeText(  
 requireContext(),  
 "Уже зарегистрирован один библиотекарь",  
 Toast.*LENGTH\_SHORT* ).show()  
 } else {  
 val user = User(  
 lastName = lastName,  
 firstName = firstName,  
 email = email,  
 login = login,  
 password = password,  
 role = role  
 )  
 db.userDao().insert(user)  
 Toast.makeText(  
 requireContext(),  
 "Регистрация успешна",  
 Toast.*LENGTH\_SHORT* ).show()  
 }  
 } else {  
 val user = User(  
 lastName = lastName,  
 firstName = firstName,  
 email = email,  
 login = login,  
 password = password,  
 role = role  
 )  
 db.userDao().insert(user)  
 Toast.makeText(  
 requireContext(),  
 "Регистрация успешна",  
 Toast.*LENGTH\_SHORT* ).show()  
 }  
 **}** }  
 **}** }  
  
 }  
 view.findViewById<ImageView>(R.id.*onLog*).setOnClickListener **{** val loginFragment = LoginFragment()  
 *parentFragmentManager*.beginTransaction()  
 .replace(R.id.*fragmentContainer*, loginFragment)  
 .addToBackStack(null)  
 .commit()  
 **}  
 }** return view  
 }

}

**StudentActivity**

class StudentActivity : AppCompatActivity() {  
  
 private lateinit var recyclerView: RecyclerView  
 private lateinit var recomend: RecyclerView  
 private lateinit var booksAdapter: BooksAdapter  
 private lateinit var recmendAdapter: BooksAdapter  
 private lateinit var db: AppDatabase  
 private val booksList = *mutableListOf*<Books>()  
 private val recomendList = *mutableListOf*<Books>()  
 private lateinit var database: AppDatabase  
 private lateinit var removeButton: Button  
  
 @SuppressLint("MissingInflatedId")  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_student*)  
  
 db = Room.databaseBuilder(*applicationContext*, AppDatabase::class.*java*, "user-db").build()  
 database = Room.databaseBuilder(this, AppDatabase::class.*java*, "books.db").build()  
  
 recyclerView = findViewById(R.id.*libView*)  
 recomend = findViewById(R.id.*recView*)  
  
 recyclerView.*layoutManager* = LinearLayoutManager(this)  
 recomend.*layoutManager* = LinearLayoutManager(this)  
  
 booksAdapter = BooksAdapter(booksList, **{** book **-> }**, **{** book **-> }**)  
 recmendAdapter = BooksAdapter(recomendList, **{** book **-> }**, **{** book **-> }**)  
  
 recyclerView.*adapter* = booksAdapter  
 recomend.*adapter* = recmendAdapter  
  
 removeButton = findViewById(R.id.*removeButton*)  
  
 removeButton.setOnClickListener **{** booksAdapter.removeSelectedBook(database.bookDao(), this.*lifecycleScope*)  
 recmendAdapter.removeSelectedBook(database.bookDao(), this.*lifecycleScope*)  
 **}** this.findViewById<ImageView>(R.id.*onLog*).setOnClickListener **{** val intent = Intent(this, RegAuto::class.*java*)  
 startActivity(intent)  
 **}** loadBooks()  
 }  
  
 private fun loadBooks() {  
 *lifecycleScope*.*launch* **{** val bookDao = database.bookDao()  
 val savedBooks = bookDao.getAllBooks()  
 booksList.clear()  
 recomendList.clear()  
  
 savedBooks.*forEach* **{** book **->** if (book.recom == true) {  
 recomendList.add(book)  
 } else {  
 booksList.add(book)  
 }  
 **}** booksAdapter.notifyDataSetChanged()  
 recmendAdapter.notifyDataSetChanged()  
 **}** }  
}

**TeacherActivity**

class TeacherActivity : AppCompatActivity() {  
  
 private lateinit var recyclerView: RecyclerView  
 private lateinit var userAdapter: UserAdapter  
 private lateinit var db: AppDatabase  
  
 @SuppressLint("MissingInflatedId")  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_teacher*)  
  
 db = Room.databaseBuilder(*applicationContext*, AppDatabase::class.*java*, "user-db").build()  
  
 recyclerView = findViewById(R.id.*recyclerView*)  
 recyclerView.*layoutManager* = LinearLayoutManager(this)  
  
 loadStudents()  
  
 findViewById<Button>(R.id.*registerButton*).setOnClickListener **{** SaveMeGod()  
 **}** findViewById<Button>(R.id.*lib*).setOnClickListener **{** val intent = Intent(this, TeacherBook::class.*java*)  
 startActivity(intent)  
 **}** this.findViewById<ImageView>(R.id.*onLog*).setOnClickListener **{** val intent = Intent(this, RegAuto::class.*java*)  
 startActivity(intent)  
 **}** }  
  
 private fun SaveMeGod() {  
 *lifecycleScope*.*launch* **{** val studentCount = db.userDao().getStudentCount()  
 if (studentCount >= 30) {  
 Toast.makeText(this@TeacherActivity, "Достигнуто максимальное количество студентов", Toast.*LENGTH\_SHORT*).show()  
 return@launch  
 }  
  
  
 val lastName = findViewById<EditText>(R.id.*lastNameEditText*).*text*.toString()  
 val firstName = findViewById<EditText>(R.id.*firstNameEditText*).*text*.toString()  
 val email = findViewById<EditText>(R.id.*emailEditText*).*text*.toString()  
 val login = findViewById<EditText>(R.id.*loginEditText*).*text*.toString()  
 val password = findViewById<EditText>(R.id.*passwordEditText*).*text*.toString()  
  
 if (lastName.*isEmpty*() || firstName.*isEmpty*() || email.*isEmpty*() || login.*isEmpty*() || password.*isEmpty*()) {  
 Toast.makeText(this@TeacherActivity, "Пожалуйста, заполните все поля", Toast.*LENGTH\_SHORT*).show()  
 } else {  
 val emailPattern = Regex("^[A-Za-z0-9.\_%+-]+@[A-Za-z0-9.-]+\\.[A-Z|a-z]{2,}$")  
 if (!emailPattern.matches(email)) {  
 Toast.makeText(this@TeacherActivity, "Введите корректный адрес электронной почты", Toast.*LENGTH\_SHORT*).show()  
 } else if (password.length < 8) {  
 Toast.makeText(this@TeacherActivity, "Пароль должен содержать не менее 8 символов", Toast.*LENGTH\_SHORT*).show()  
 } else {  
 val user = User(  
 lastName = lastName,  
 firstName = firstName,  
 email = email,  
 login = login,  
 password = password,  
 role = "Студент"  
 )  
  
 db.userDao().insert(user)  
 Toast.makeText(this@TeacherActivity, "Регистрация успешна", Toast.*LENGTH\_SHORT*).show()  
 loadStudents()  
 }  
 }  
 **}** }  
  
 private fun loadStudents() {  
 *lifecycleScope*.*launch* **{** val students = db.userDao().getStudents()  
 userAdapter = UserAdapter(students) **{** student **->** deleteStudent(student)  
 **}** recyclerView.*adapter* = userAdapter  
 **}** }  
  
 private fun deleteStudent(student: User) {  
 *lifecycleScope*.*launch* **{** db.userDao().delete(student)  
 userAdapter.removeStudent(student)  
 Toast.makeText(this@TeacherActivity, "Студент удалён", Toast.*LENGTH\_SHORT*).show()  
 **}** }  
  
}

**TeacherBook**

class TeacherBook : AppCompatActivity() {  
  
 private lateinit var recyclerView: RecyclerView  
 private lateinit var recomend: RecyclerView  
 private lateinit var booksAdapter: BooksAdapter  
 private lateinit var recmendAdapter: BooksAdapter  
 private lateinit var db: AppDatabase  
 private val booksList = *mutableListOf*<Books>()  
 private val recomendList = *mutableListOf*<Books>()  
 private lateinit var database: AppDatabase  
  
 @SuppressLint("MissingInflatedId")  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_teacher\_book*)  
  
 db = Room.databaseBuilder(*applicationContext*, AppDatabase::class.*java*, "user-db").build()  
 database = Room.databaseBuilder(this, AppDatabase::class.*java*, "books.db").build()  
  
 recyclerView = findViewById(R.id.*libView*)  
 recomend = findViewById(R.id.*recView*)  
  
 recyclerView.*layoutManager* = LinearLayoutManager(this)  
 recomend.*layoutManager* = LinearLayoutManager(this)  
  
 booksAdapter = BooksAdapter(booksList, **{** book **-> }**, **{** book **-> }**)  
 recmendAdapter = BooksAdapter(recomendList, **{** book **-> }**, **{** book **-> }**)  
  
 recyclerView.*adapter* = booksAdapter  
 recomend.*adapter* = recmendAdapter  
  
 loadBooks()  
  
 this.findViewById<ImageView>(R.id.*onLog*).setOnClickListener **{** val intent = Intent(this, TeacherActivity::class.*java*)  
 startActivity(intent)  
 **}** }  
  
 private fun loadBooks() {  
 *lifecycleScope*.*launch* **{** val bookDao = database.bookDao()  
 val savedBooks = bookDao.getAllBooks()  
 booksList.clear()  
 recomendList.clear()  
  
 savedBooks.*forEach* **{** book **->** if (book.recom == true) {  
 recomendList.add(book)  
 } else {  
 booksList.add(book)  
 }  
 **}** booksAdapter.notifyDataSetChanged()  
 recmendAdapter.notifyDataSetChanged()  
 **}** }  
  
}

**User**

@Entity(tableName = "User")  
data class User(  
 @PrimaryKey(autoGenerate = true) val id: Int = 0,  
 val lastName: String,  
 val firstName: String,  
 val email: String,  
 val login: String,  
 val password: String,  
 val role: String  
)

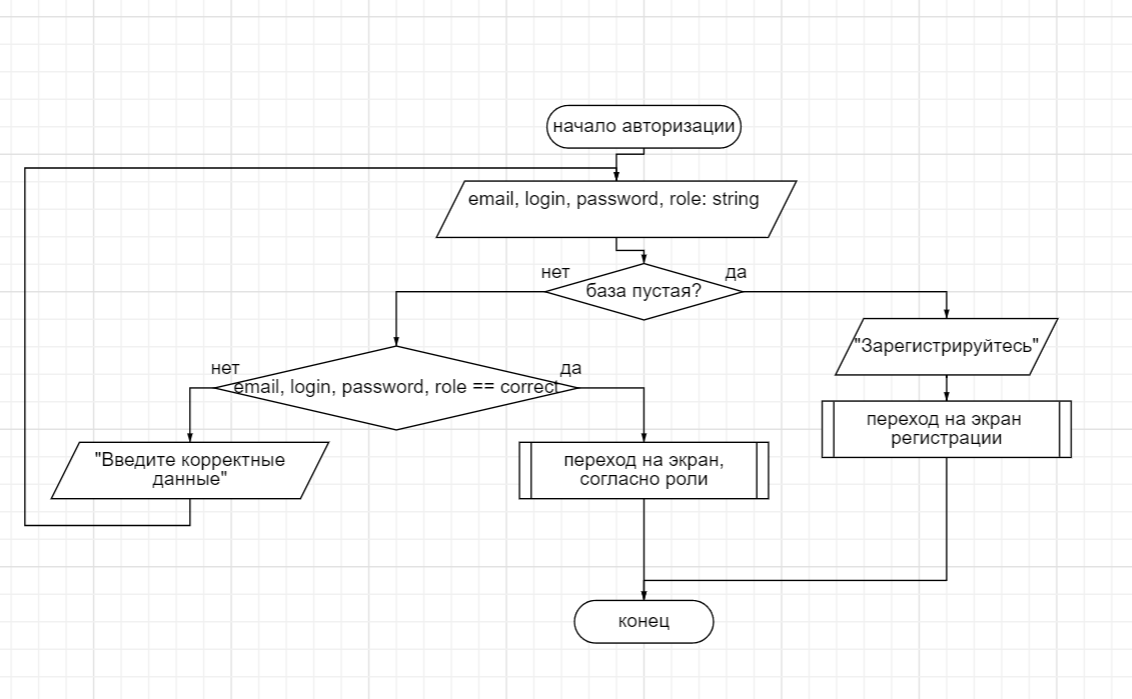
**UserAdapter**

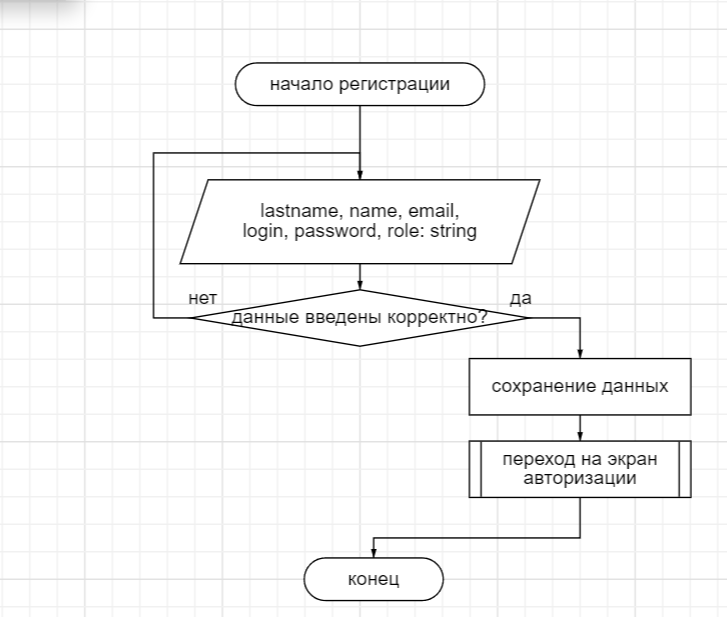
class UserAdapter(private var studentList: List<User>, private val onDeleteClick: (User) -> Unit) : RecyclerView.Adapter<UserAdapter.UserViewHolder>() {  
  
  
 class UserViewHolder(view: View) : RecyclerView.ViewHolder(view) {  
  
 val firstName: TextView = view.findViewById(R.id.*fam*)  
 val lastName: TextView = view.findViewById(R.id.*name*)  
 val email: TextView = view.findViewById(R.id.*mail*)  
 fun bind(student: User) {  
 lastName.*text* = student.lastName  
 firstName.*text* = student.firstName  
 }  
 }  
  
 override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): UserViewHolder {  
 val view = LayoutInflater.from(parent.*context*).inflate(R.layout.*item\_student*, parent, false)  
 return UserViewHolder(view)  
 }  
  
 override fun onBindViewHolder(holder: UserViewHolder, position: Int) {  
 val student = studentList[position]  
 holder.bind(student)  
  
 holder.itemView.setOnClickListener **{** onDeleteClick(student)  
 **}** }  
 fun removeStudent(student: User) {  
 val updatedList = studentList.*toMutableList*()  
 updatedList.remove(student)  
 studentList = updatedList  
 notifyDataSetChanged()  
 }  
  
 override fun getItemCount() = studentList.size  
}

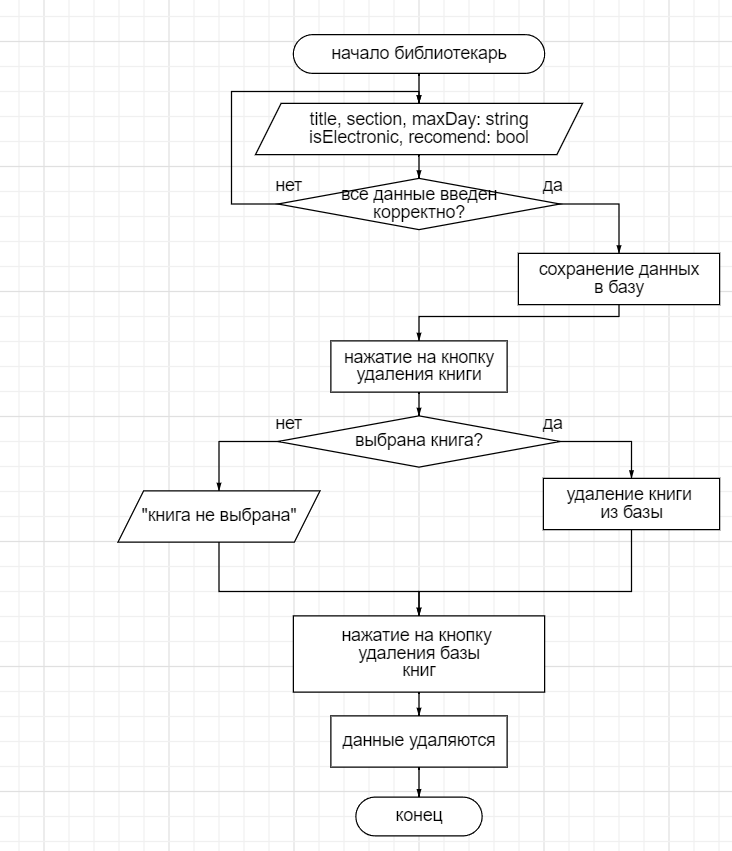
**UserDao**

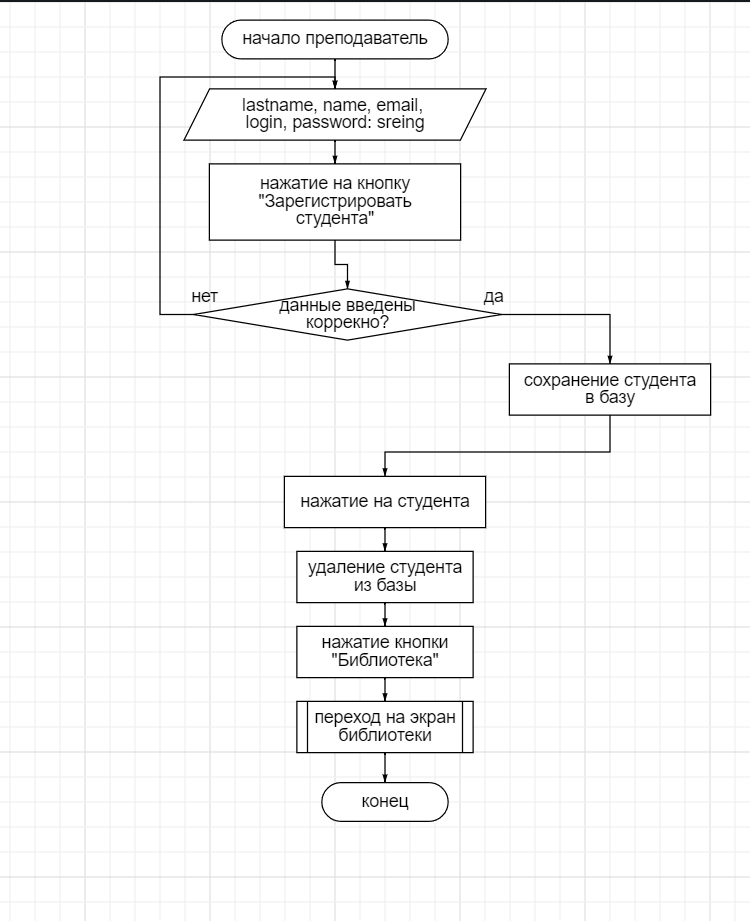
@Dao  
interface UserDao {  
 @Insert  
 suspend fun insert(user: User)  
  
 @Query("SELECT \* FROM User WHERE email = :email AND login = :login AND password = :password AND role = :role LIMIT 1")  
 suspend fun getUser(email: String, login: String, password: String, role: String): User?  
  
 @Query("SELECT COUNT(\*) FROM User")  
 suspend fun getUserCount(): Int  
 @Query("SELECT \* FROM User WHERE role = 'Студент'")  
 suspend fun getStudents(): List<User>  
  
 @Delete  
 suspend fun delete(user: User)  
  
 @Query("SELECT \* FROM user WHERE login = :login OR email = :email LIMIT 1")  
 suspend fun findByLoginOrEmail(login: String, email: String): User?  
  
 @Query("SELECT COUNT(\*) FROM User WHERE role = :role")  
 suspend fun countUsersByRole(role: String): Int  
  
 @Query("SELECT COUNT(\*) FROM User WHERE role = 'Студент'")  
 suspend fun getStudentCount(): Int  
  
}

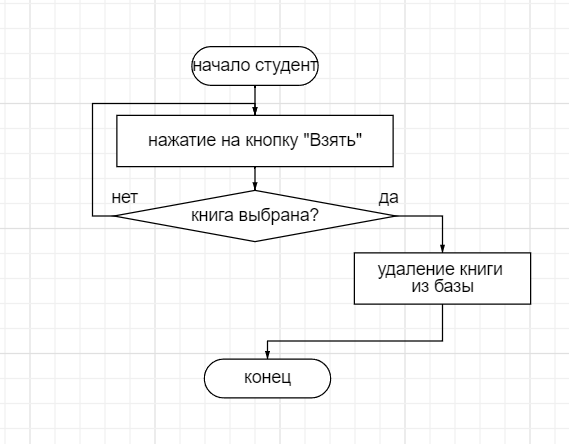
**Алгоритм решения:**











**Используемые библиотеки:**

|  |  |
| --- | --- |
| android.annotation.SuppressLint | Эта аннотация используется для подавления предупреждений компилятора о потенциальных проблемах в коде |
| android.graphics.Color | Этот класс предоставляет константы и методы для работы с цветами |
| androidx.appcompat.app.AppCompatActivity | AppCompatActivity — это базовый класс для активностей, который обеспечивает поддержку старых версий Android и позволяет использовать компоненты Material Design |
| android.os.Bundle | Класс Bundle используется для передачи данных между активностями и сохранения состояния активности |
| android.widget.GridLayout | GridLayout — это контейнер, который позволяет размещать дочерние элементы в виде сетки |
| android.widget.ImageView | Этот класс представляет элемент пользовательского интерфейса, который отображает изображения. |
| android.widget.LinearLayout | LinearLayout — это контейнер, который размещает дочерние элементы в одном направлении (горизонтально или вертикально) |
| android.widget.TextView | Этот класс представляет собой элемент пользовательского интерфейса для отображения текста |
| com.google.gson.Gson | Gson — это библиотека для работы с JSON в Java и Kotlin. Она позволяет легко сериализовать и десериализовать объекты, преобразуя их в JSON и обратно |
| com.squareup.picasso.Picasso | Picasso — это библиотека для загрузки и кэширования изображений для Android. Она упрощает процесс загрузки изображений из интернета в ImageView |
| okhttp3 | OkHttp — это мощный HTTP-клиент для приложений Android и Java. Он используется для отправки и получения сетевых запросов на основе HTTP |

**Используемые инструменты:**

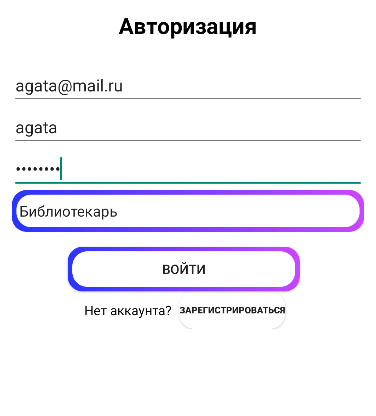
Android Stusio, Kotlin

**Описание пользовательского интерфейса:**

После входа в приложение пользователя встречает экран авторизации:



Если у пользователя уже есть аккаунт, то он вводит свою почту, логин и пароль, затем выбирает роль и нажимает кнопку «Войти»



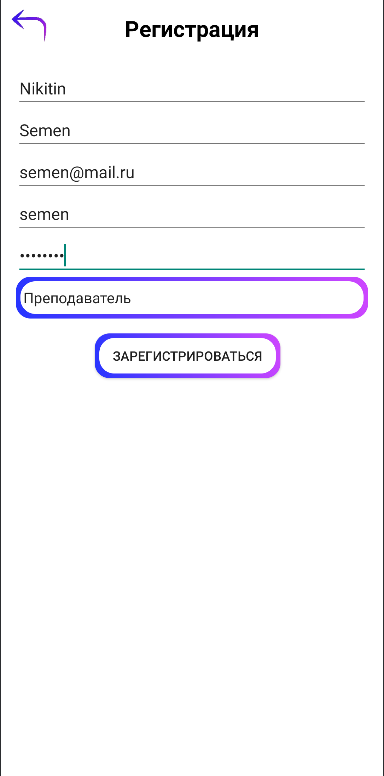
Если в базе данных совсем нет аккаунтов, то пользователь увидит уведомление:



Если аккаунты есть, но пользователь ввёл неверные данные или выбрал не свою роль, то ему будет показано сообщение об ошибке:

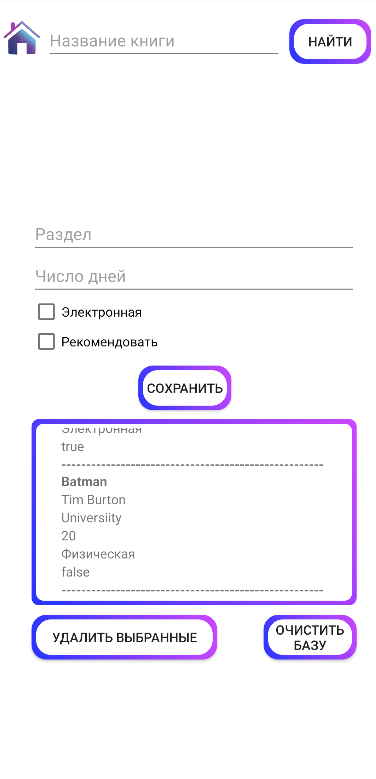


Если у пользователя нет аккаунта, то он должен нажать на кнопку «Зарегистрироваться», после чего он перейдёт на экран, где должен ввести свои данные (если какие-то данные будут введены неправильно, пользователь увидит сообщение об ошибке):

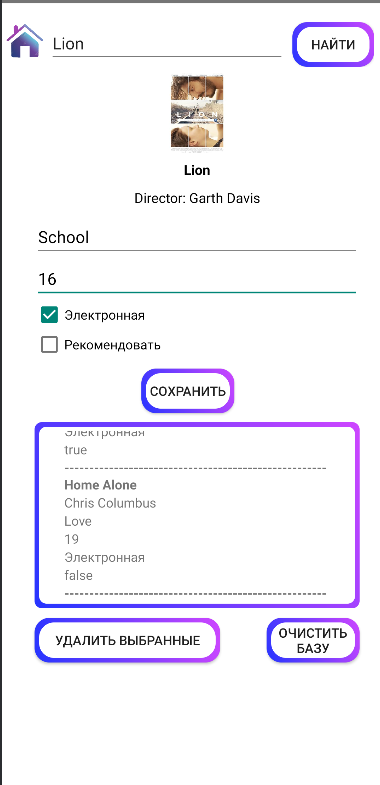


Когда пользователь зарегистрируется и введёт свои данные корректно, то в зависимости от роли он перейдёт на определённые экран:

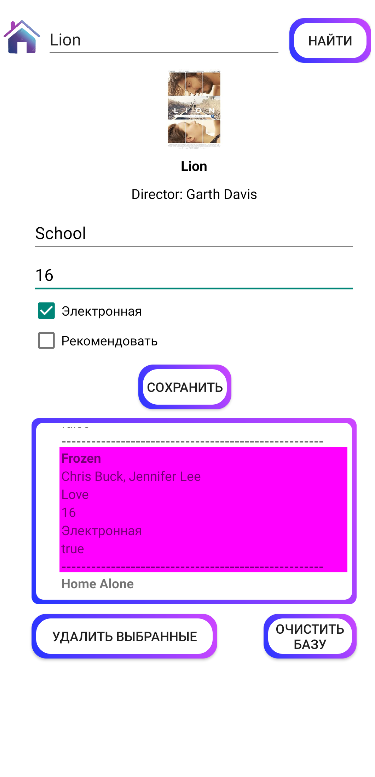
Библиотекарь:



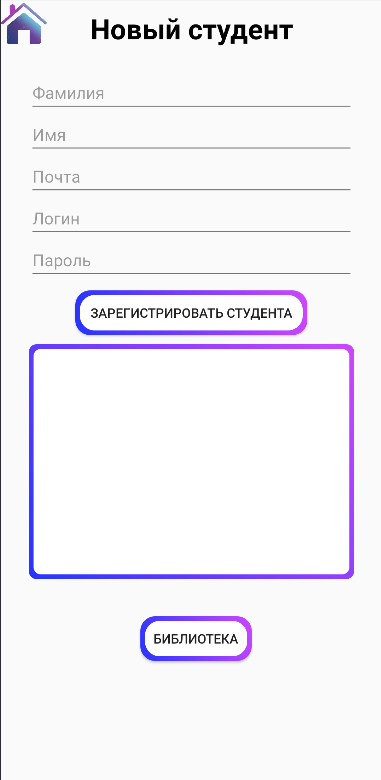
Пользователь должен ввести книгу, а затем нажать кнопку «Найти», после чего через удалённую базу (API) будет выведена обложка, название книги и её автор. Пользователь же должен ввести дополнительные данные, нажать «Сохранить» и книга сохранится в базу:



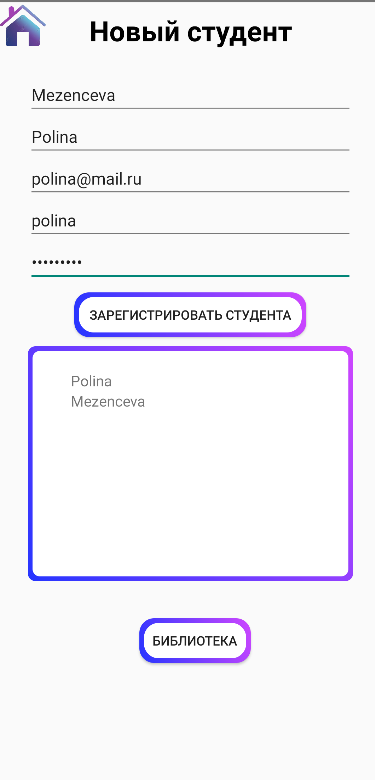
Также пользователь может удалить выделенную книгу или удалить всю книжную базу полностью:



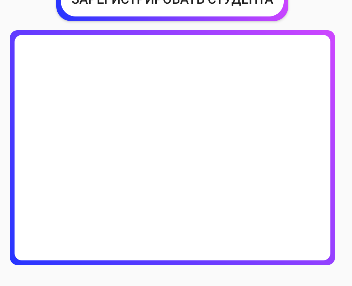
Преподаватель:



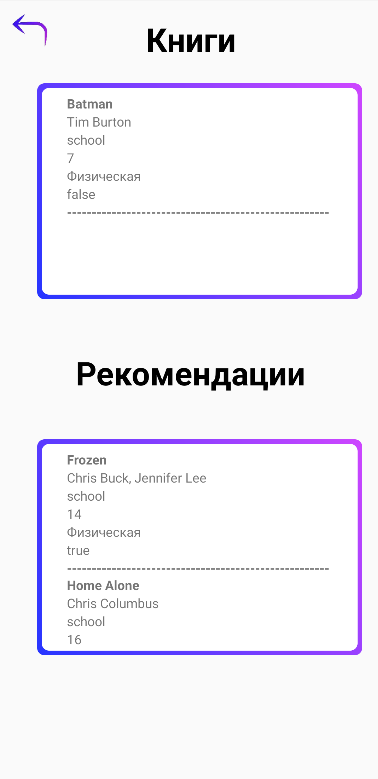
Преподаватель может зарегистрировать нового студента:



Или удалить его нажатием на его аккаунт:



А также посмотреть книги, хранящиеся в библиотеке, нажав на кнопку «Библиотека»:



Студент:



Выбрав книгу пользователь может выбрать книгу, а затем взять её, нажав кнопку «Взять»:

