НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО”

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

Лабораторна робота №4

з дисципліни

“Програмування мобільних систем”

Виконав:

студент групи ІО-82

ЗК ІО-8206

Востриков Нікіта

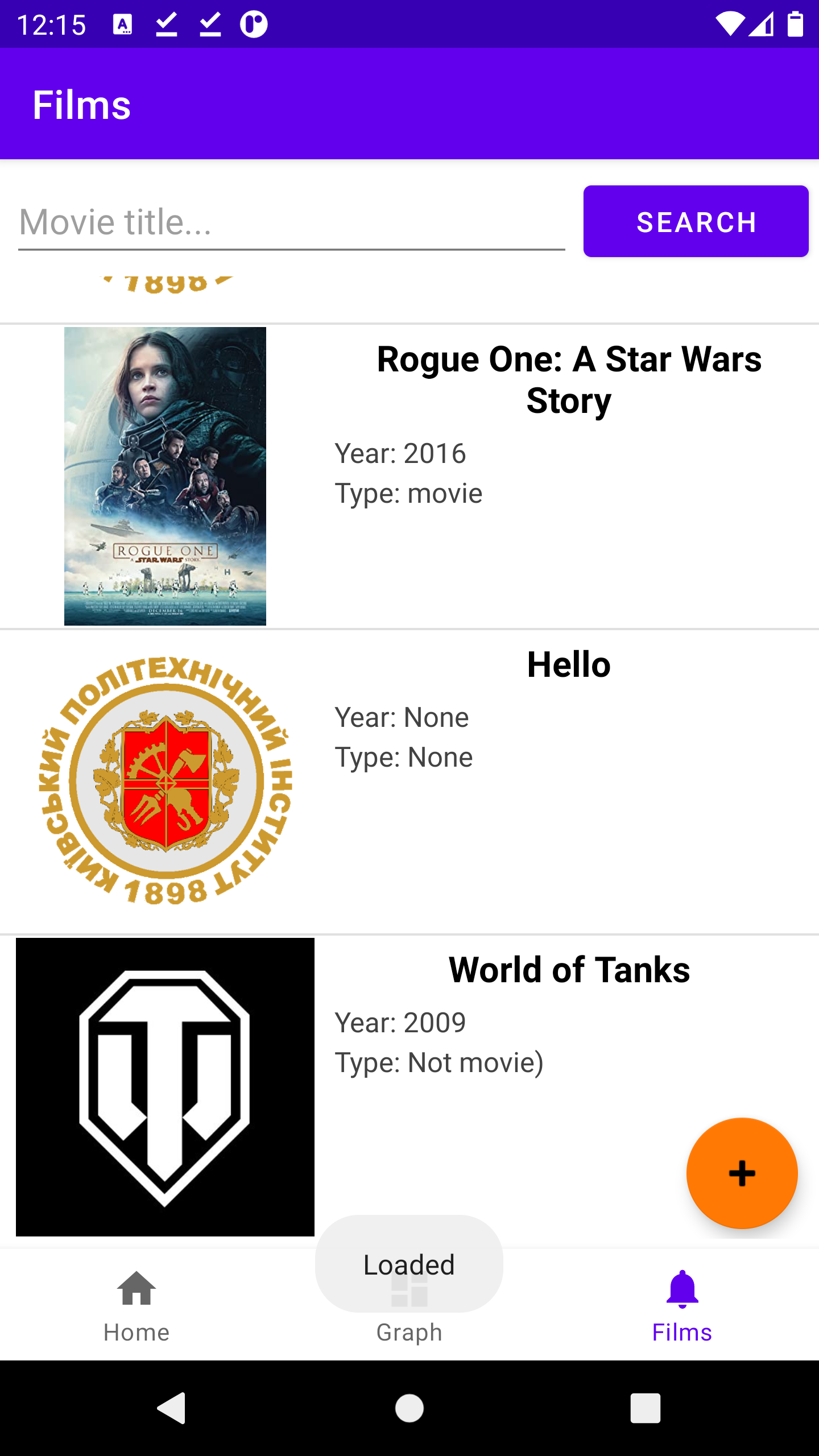
Перевірив:

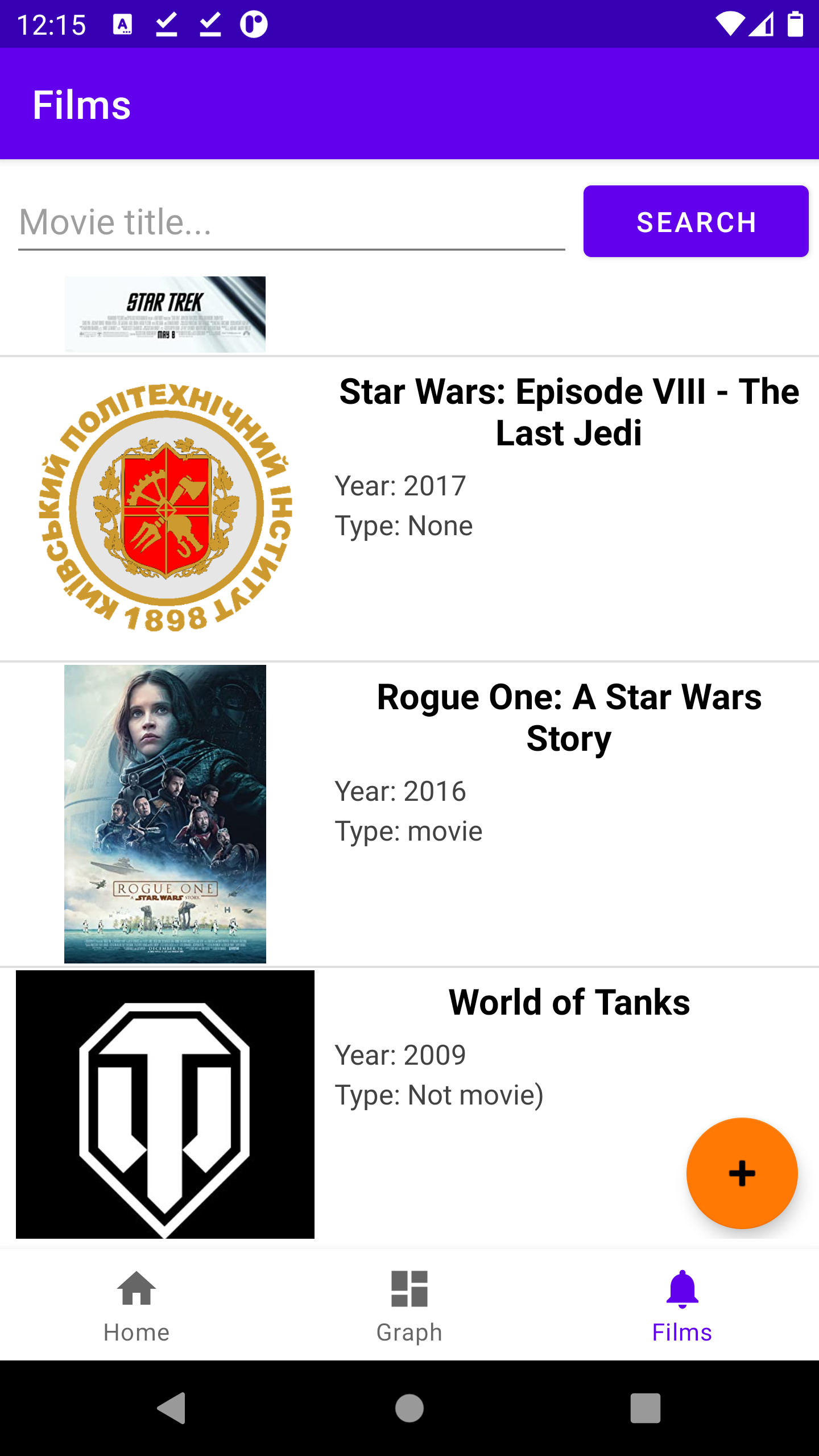
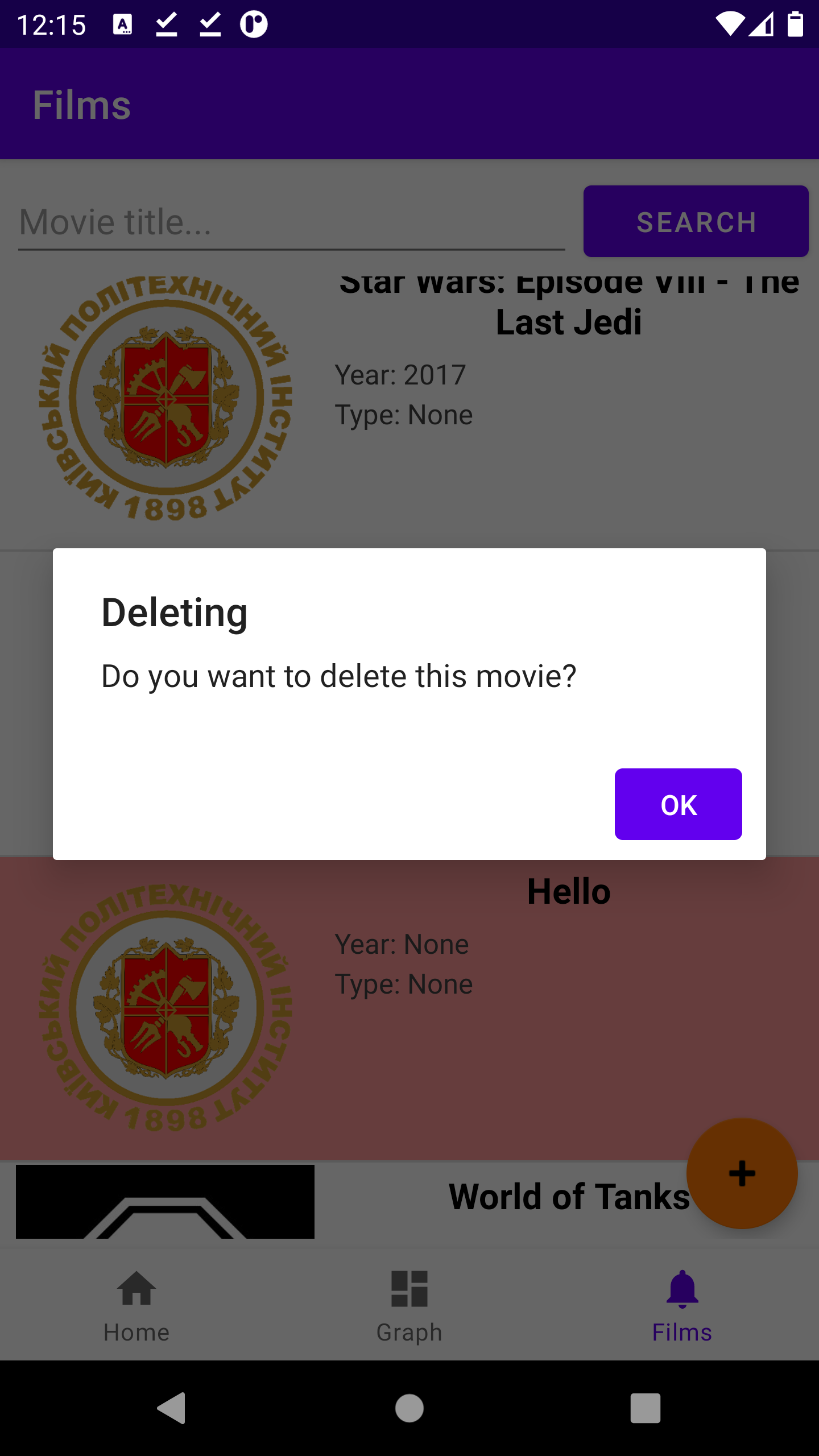
Шульга М. В.

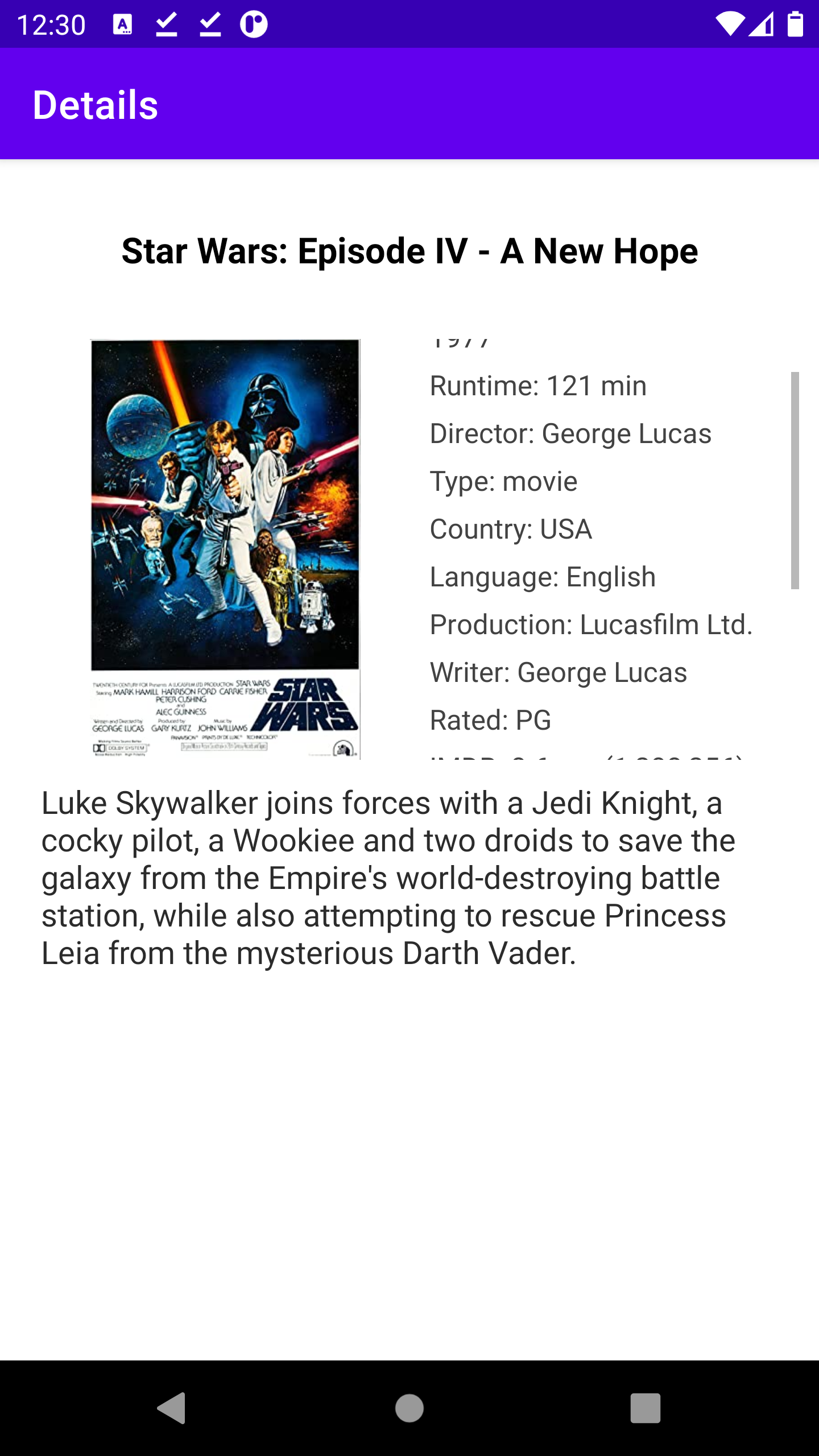
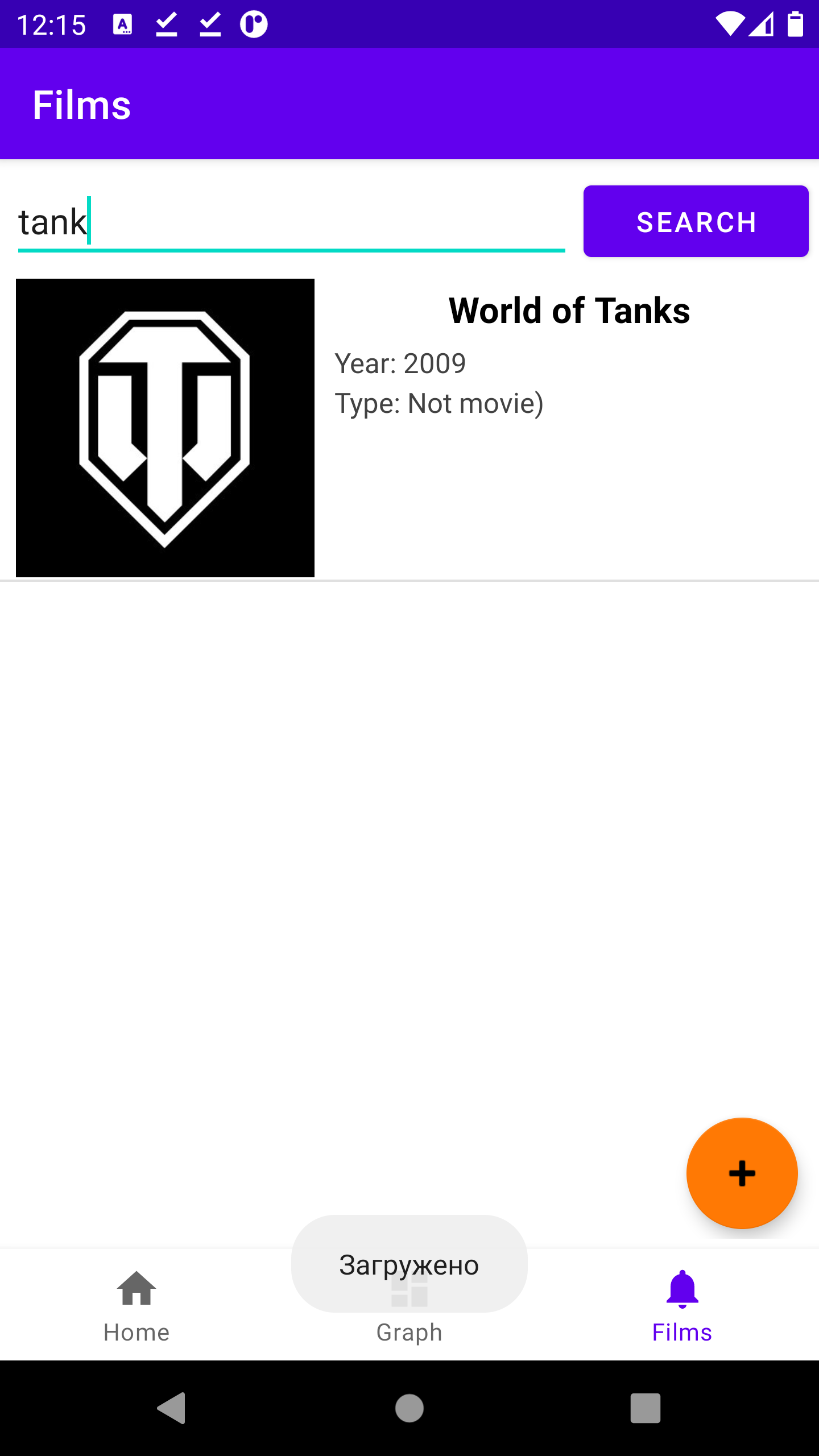
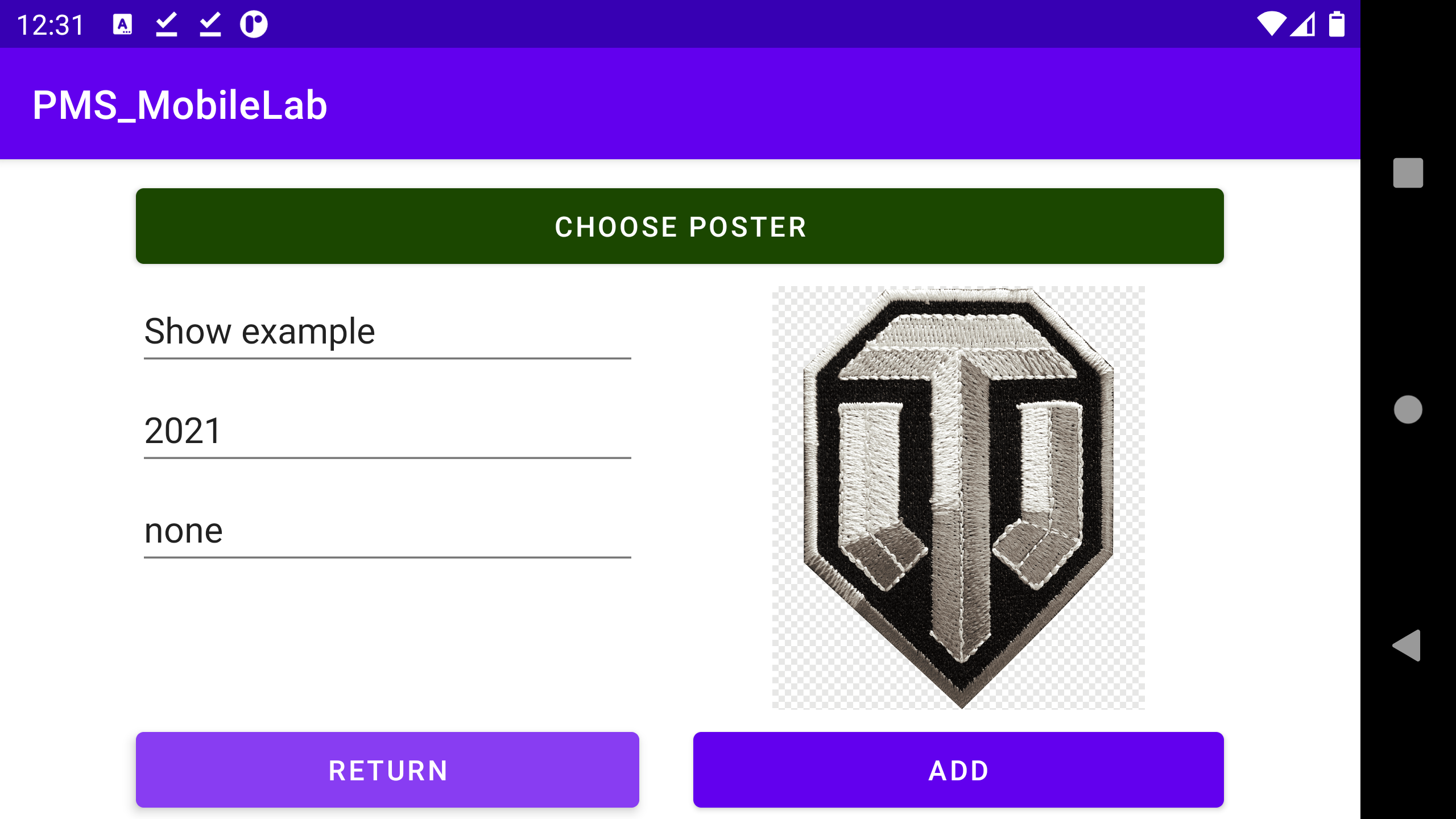
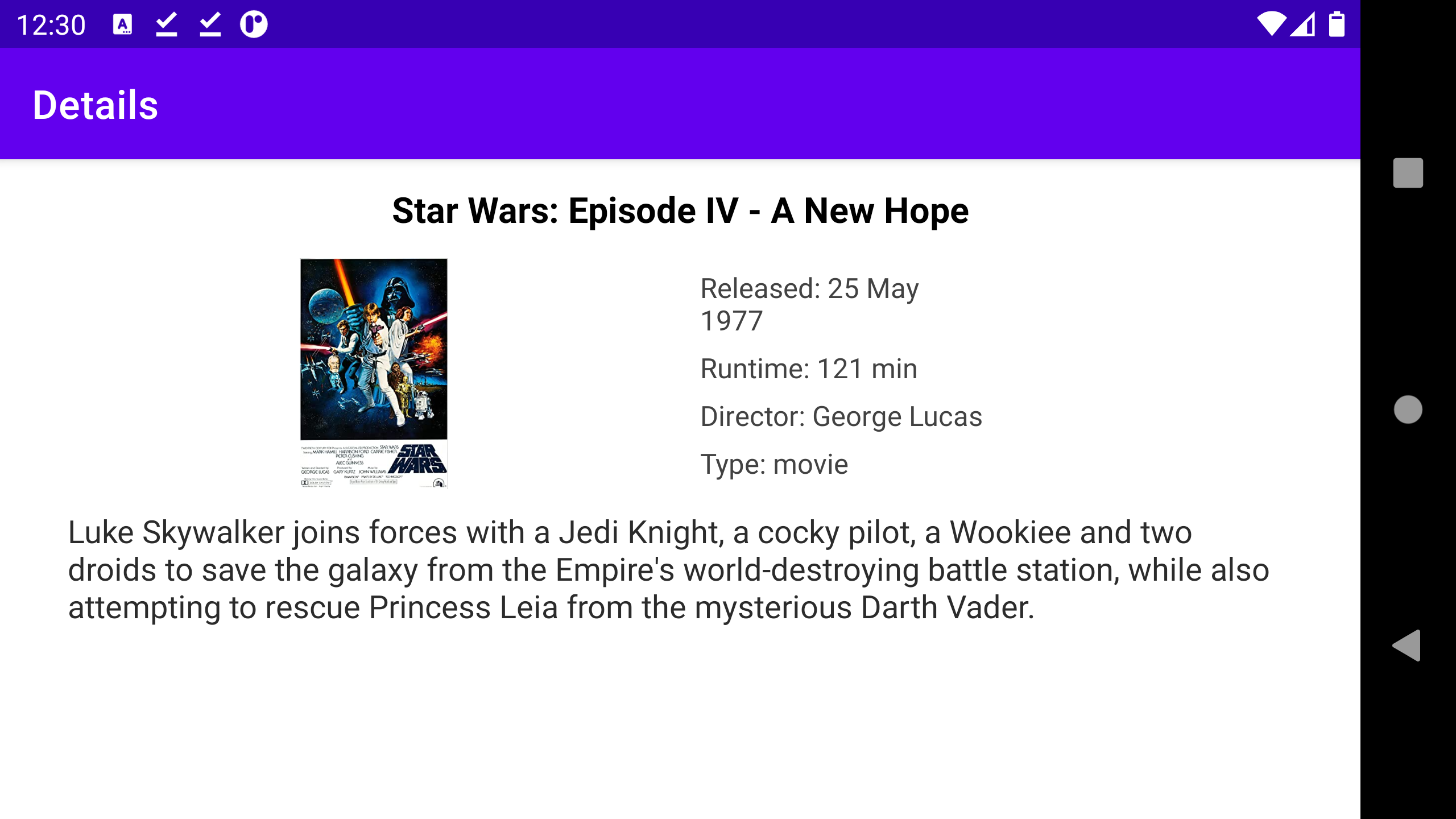
Миколаїв 2021

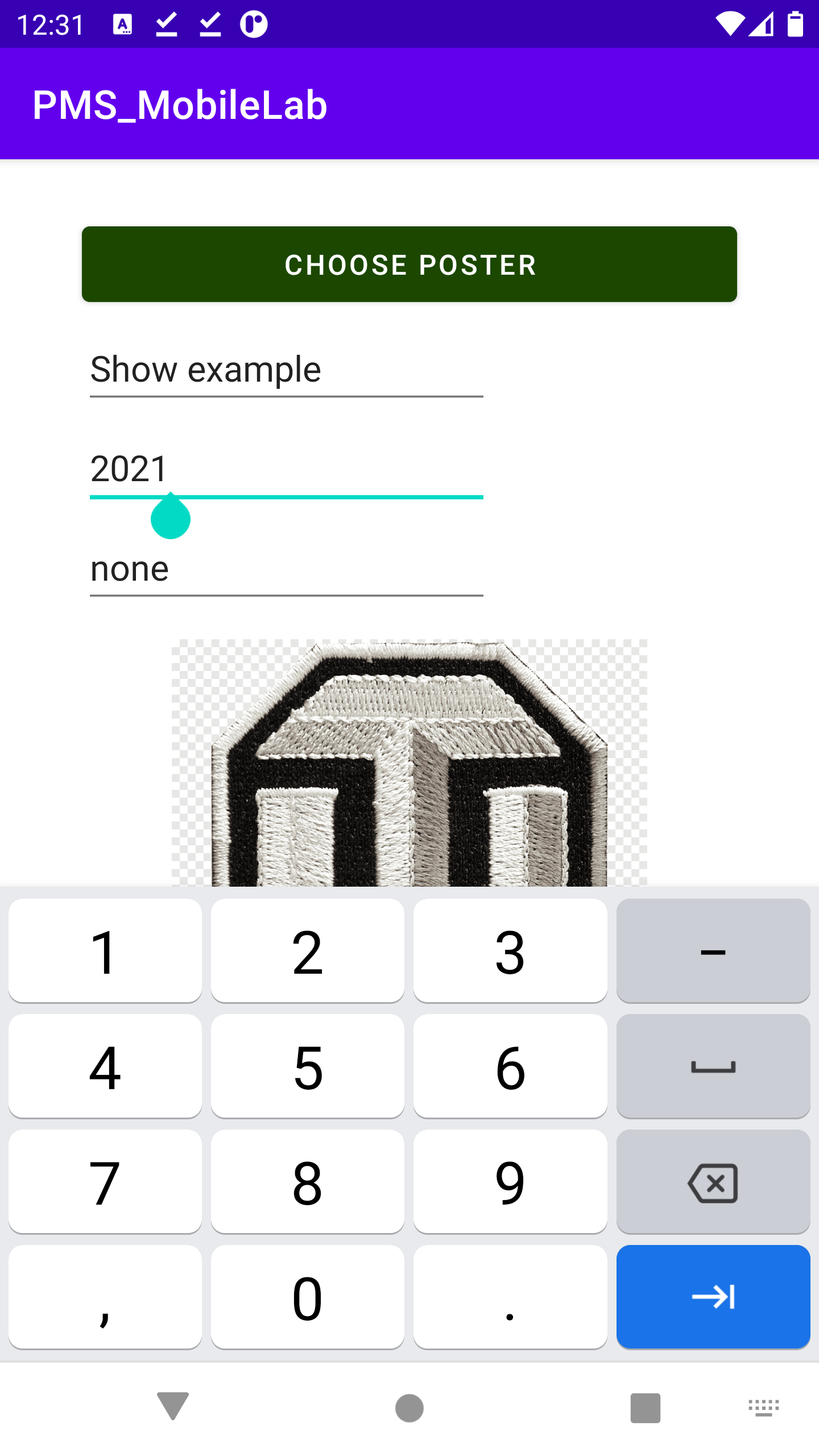
**Варіант = 8206%2 + 1 = 1**

Для розробки використовується IDE Android Studio та мова програмування Java.

**Скріншоти роботи додатка**



****

**Лістинг коду**

**MainActivity.java**

package ua.kpi.comsys.IO8206;  
  
import android.os.Bundle;  
import android.widget.Toast;  
  
import com.google.android.material.bottomnavigation.BottomNavigationView;  
  
import androidx.appcompat.app.AppCompatActivity;  
import androidx.navigation.NavController;  
import androidx.navigation.Navigation;  
import androidx.navigation.ui.AppBarConfiguration;  
import androidx.navigation.ui.NavigationUI;  
  
public class MainActivity extends AppCompatActivity {  
 BottomNavigationView navView;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);  
 navView = findViewById(R.id.*nav\_view*);  
 *// Passing each menu ID as a set of Ids because each  
 // menu should be considered as top level destinations.* AppBarConfiguration appBarConfiguration = new AppBarConfiguration.Builder(  
 R.id.*navigation\_home*, R.id.*navigation\_dashboard*, R.id.*navigation\_notifications*)  
 .build();  
 NavController navController = Navigation.*findNavController*(this, R.id.*nav\_host\_fragment*);  
 NavigationUI.*setupActionBarWithNavController*(this, navController, appBarConfiguration);  
 NavigationUI.*setupWithNavController*(navView, navController);  
 }  
  
*// @Override  
// protected void onDestroy() {  
// super.onDestroy();  
// System.out.println("DESTROY1");  
// Toast.makeText(navView.getContext(), "Good luck :)", Toast.LENGTH\_LONG).show();  
// }*}

**DashboardFragment.java**

package ua.kpi.comsys.IO8206.ui.dashboard;  
  
import android.graphics.Color;  
import android.media.audiofx.AudioEffect;  
import android.os.Bundle;  
import android.view.LayoutInflater;  
import android.view.View;  
import android.view.ViewGroup;  
  
import androidx.annotation.NonNull;  
import androidx.fragment.app.Fragment;  
  
import com.github.mikephil.charting.charts.PieChart;  
import com.github.mikephil.charting.components.Description;  
import com.github.mikephil.charting.data.PieData;  
import com.github.mikephil.charting.data.PieDataSet;  
import com.github.mikephil.charting.data.PieEntry;  
import com.jjoe64.graphview.GraphView;  
import com.jjoe64.graphview.series.DataPoint;  
import com.jjoe64.graphview.series.LineGraphSeries;  
  
import java.util.ArrayList;  
import java.util.Arrays;  
import java.util.List;  
  
import ua.kpi.comsys.IO8206.R;  
  
public class DashboardFragment extends Fragment {  
 LineGraphSeries<DataPoint> series1;  
  
  
*// private DashboardViewModel dashboardViewModel;* public View onCreateView(@NonNull LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {  
 View root = inflater.inflate(R.layout.*fragment\_sec\_tab*, container, false);  
  
 int dots = 100;  
 double min = -5.0, max = 5.0, currentPointX=min, currentPointY, step;  
 dots -= 1;  
 DataPoint[] dataPoint = new DataPoint[dots+1];  
 step = (max-min)/dots;  
  
 for (int i = 0; i <= dots; i++) {  
 currentPointY = currentPointX\*currentPointX;  
 dataPoint[i] = new DataPoint(currentPointX, currentPointY);  
 currentPointX += step;  
 }  
  
 GraphView graph = (GraphView) root.findViewById(R.id.*graph*); *// график* PieChart pieChart = root.findViewById(R.id.*pieChart*); *// круговая диаграмма* graph.setFocusable(true);  
 series1 = new LineGraphSeries<>(dataPoint);  
 System.*out*.println(Arrays.*toString*(dataPoint));  
 graph.addSeries(series1);  
  
 graph.getViewport().setMinX(min-1); *// установка границ* graph.getViewport().setMaxX(max+1);  
 graph.getViewport().setMaxY(max\*max+5);  
  
 graph.getViewport().setXAxisBoundsManual(true);  
 graph.getViewport().setYAxisBoundsManual(true);  
  
  
 pieChart.setUsePercentValues(true);  
 Description desc = new Description();  
 desc.setText("Variant 5");  
 desc.setTextSize(20f);  
  
 List<PieEntry> value = new ArrayList<>();  
 value.add(new PieEntry(35f, "green"));  
 value.add(new PieEntry(40f, "yellow"));  
 value.add(new PieEntry(25f, "red"));  
  
 int[] colors = {getResources().getColor(R.color.*green*), getResources().getColor(R.color.*yellow*), getResources().getColor(R.color.*red*)};  
  
 PieDataSet pieDataSet = new PieDataSet(value, "Chart");  
 PieData pieData = new PieData(pieDataSet);  
 pieChart.setData(pieData);  
  
 pieDataSet.setValueTextSize(15f);  
 pieDataSet.setColors(colors);  
 pieChart.setDescription(desc);  
  
 return root;  
 }  
}

**FilmsList.java**

package ua.kpi.comsys.IO8206.ui.films;  
  
import android.app.AlertDialog;  
import android.content.Context;  
import android.content.DialogInterface;  
import android.content.Intent;  
import android.graphics.Bitmap;  
import android.graphics.BitmapFactory;  
import android.os.Bundle;  
import android.view.LayoutInflater;  
import android.view.View;  
import android.view.ViewGroup;  
import android.widget.AdapterView;  
import android.widget.ArrayAdapter;  
import android.widget.Button;  
import android.widget.EditText;  
import android.widget.ImageView;  
import android.widget.ListView;  
import android.widget.TextView;  
import android.widget.Toast;  
  
import androidx.annotation.NonNull;  
import androidx.annotation.Nullable;  
import androidx.fragment.app.Fragment;  
  
import com.google.android.material.floatingactionbutton.FloatingActionButton;  
  
import java.io.File;  
import java.io.FileInputStream;  
import java.io.FileWriter;  
import java.io.IOException;  
import java.io.InputStream;  
import java.util.ArrayList;  
import java.util.List;  
  
import ua.kpi.comsys.IO8206.AddFilmActivity;  
import ua.kpi.comsys.IO8206.Film;  
import ua.kpi.comsys.IO8206.JsonHelper;  
import ua.kpi.comsys.IO8206.R;  
import ua.kpi.comsys.IO8206.ui.FilmDetail;  
  
public class FilmsList extends Fragment {  
 private List<Film> films;  
 private List<Film> searchedFilms = new ArrayList<>();  
 private List<Film> filmsToShow = new ArrayList<>();  
 private FilmAdapter adapter;  
 ListView listView;  
 Boolean elemAddOnStop = false; *// добавление элемента и необходимо обновить список* Boolean searchMode = false; *// режим поиска по списку* String userFileMovie = "movieslistuser.txt"; *// стандартное значение, которое заменится* String freeSpace = (new String(new char[100]).replace("\0", "\t"));  
 Film removedElement=null;  
  
  
 @Override  
 public void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
*// if(isAdded()){  
// userFileMovie = getResources().getString(R.string.user\_films\_list); // берём имя файла из ресурсов  
// }* }  
  
 @Override  
 public void onStop() {  
 super.onStop();  
*// adapter.notifyDataSetChanged();* }  
  
*// @Override  
// public void onDestroy() {  
// super.onDestroy();  
// Toast.makeText(getContext(), "Good luck :)", Toast.LENGTH\_LONG).show();  
// }* @Override  
 public void onPause() {  
 super.onPause();  
 }  
  
 @Override  
 public void onResume() {  
 super.onResume();  
 if(elemAddOnStop){ *// если до этого был стоп(открыто окно добавления)* requireActivity().recreate(); *// обновить ЛистВью* elemAddOnStop = false;  
 }  
 }  
  
 public View onCreateView(@NonNull LayoutInflater inflater,  
 ViewGroup container, Bundle savedInstanceState) {  
  
 View root = inflater.inflate(R.layout.*fragment\_third\_tab*, container, false);  
 JsonHelper jsonHelper = new JsonHelper(R.raw.*movieslist*);  
 jsonHelper.*setFileUserName*(userFileMovie);  
 EditText searchRequest = root.findViewById(R.id.*filmSearchField*); *// поле поиска* Button searchBtn = root.findViewById(R.id.*buttonSearch*);  
 FloatingActionButton addFilmBtn = root.findViewById(R.id.*filmAddBtn*);  
  
 listView = root.findViewById(R.id.*filmsList*);  
 films = jsonHelper.*importFilmListFromJSON*(getContext()); *// берём фильмы из файла* if(films != null){  
 adapter = new FilmAdapter(getActivity(), R.layout.*activity\_list*, films);  
  
 listView.setAdapter(adapter);  
 Toast.*makeText*(getContext(), "Loaded", Toast.*LENGTH\_LONG*).show();  
 }  
 else{  
 Toast.*makeText*(getContext(), "Failed to get data", Toast.*LENGTH\_LONG*).show();  
 }  
  
 listView.setOnItemClickListener(new AdapterView.OnItemClickListener() { *// нажатие на элемент списка* @Override  
 public void onItemClick(AdapterView<?> parent, View itemClicked, int position,  
 long id) {  
 Toast.*makeText*(getContext(), filmsToShow.get((int)id).getTitle(),  
 Toast.*LENGTH\_SHORT*).show();  
  
 startActivity(new Intent(getContext(), FilmDetail.class).putExtra("filmImdbId", filmsToShow.get((int)id).getImdbID()));  
 }  
 });  
  
 listView.setOnItemLongClickListener(new AdapterView.OnItemLongClickListener() { *// долгое нажатие на элемент списка* @Override  
 public boolean onItemLongClick(AdapterView<?> parent, View itemClicked, int position,  
 long id) {  
 if(!searchMode) {  
 itemClicked.setBackgroundResource(R.color.*light\_red*); *// выделить элемент* try {  
 AlertDialog.Builder builder = new AlertDialog.Builder(getActivity());  
 builder.setTitle("Deleting");  
 builder.setMessage("Do you want to delete this movie?");  
 builder.setCancelable(true);  
 builder.setOnCancelListener(new DialogInterface.OnCancelListener() { *// закрыть диалог* @Override  
 public void onCancel(DialogInterface dialog) {  
 itemClicked.setBackgroundResource(R.color.*white*); *// вернуть цвет* }  
 });  
  
 builder.setPositiveButton(android.R.string.*yes*, new DialogInterface.OnClickListener() { *// Кнопка YES* @Override  
 public void onClick(DialogInterface dialog, int which) {  
 removedElement = films.remove((int) id); *// удалить выбранный элемент* adapter.notifyDataSetChanged(); *// обновить окно* jsonHelper.*exportToJSON*(getContext(), films);  
 elemAddOnStop = true;  
 dialog.dismiss(); *// Отпускает диалоговое окно* }  
  
 });  
 AlertDialog dialog = builder.create();  
 dialog.show(); *// показать диалог* } catch (Exception e) {  
 Toast.*makeText*(getContext(), "Deleting error", Toast.*LENGTH\_LONG*).show();  
 }  
 }  
 else Toast.*makeText*(getContext(), "To delete. you must leave the search mode", Toast.*LENGTH\_LONG*).show();  
 return true;  
 }  
 });  
  
 addFilmBtn.setOnClickListener(new View.OnClickListener() { *// при нажатии на кнопку "добавить"* public void onClick(View view) {  
 elemAddOnStop = true;  
 startActivity(new Intent(getContext(), AddFilmActivity.class).putExtra("moviesListId", R.raw.*movieslist*));  
 }  
 });  
  
  
 searchBtn.setOnClickListener(new View.OnClickListener() { *// при нажатии на кнопку "поиск"* public void onClick(View view) {  
 String fieldText = searchRequest.getText().toString().toLowerCase();  
 FilmAdapter adapter2;  
 searchedFilms.clear();  
  
 if (fieldText.equals("!reset")){ *// сброс пользовательский изменений* File userFile = new File(view.getContext().getFilesDir() + "/" + userFileMovie);  
  
 try(FileWriter writer = new FileWriter(userFile)){  
 jsonHelper.*setUserFileEnable*(false);  
 writer.write(jsonHelper.*getStringFromRawFile*(getContext())); *// запись в файл юзерспейса JSON`а* writer.flush();  
 }  
 catch(IOException ex){  
 ex.printStackTrace();  
 }  
 getActivity().recreate();  
 Toast.*makeText*(getContext(), "User list has been reset", Toast.*LENGTH\_LONG*).show();  
 adapter2 = new FilmAdapter(getActivity(), R.layout.*activity\_list*, films); *// адаптер с стандартным списком* }  
  
 else if(!fieldText.equals("")){  
 searchMode = true;  
 for (int i = 0; i < films.size(); i++) {  
 if(films.get(i).getTitle().toLowerCase().contains(fieldText)){ *// ищем совпадения в заголовках и добавляем в  
 // второй список фильмов, если есть совпадения* searchedFilms.add(films.get(i));  
 }  
 }  
  
 if(searchedFilms.isEmpty()){  
 Toast.*makeText*(getContext(), "Ничего не найдено :(", Toast.*LENGTH\_LONG*).show();  
 }  
 else Toast.*makeText*(getContext(), "Загружено", Toast.*LENGTH\_LONG*).show();  
 adapter2 = new FilmAdapter(getActivity(), R.layout.*activity\_list*, new ArrayList<>(searchedFilms)); *// адаптер с новыми фильмами* }  
 else {  
 searchMode = false;  
 adapter2 = new FilmAdapter(getActivity(), R.layout.*activity\_list*, films); *// адаптер с стандартным списком* };  
 listView.setAdapter(adapter2);  
 }  
 });  
 return root;  
 }  
  
 private class FilmAdapter extends ArrayAdapter<Film>{ *// свой адаптер* FilmAdapter(Context context, int textViewResourceId, List<Film> objects) {  
 super(context, textViewResourceId, objects);  
 filmsToShow = objects; *// список найденых фильмов* }  
  
 @NonNull  
 @Override  
 public View getView(int position, @Nullable View convertView, @NonNull ViewGroup parent) { *// переопределение* LayoutInflater inflater = getLayoutInflater();  
 View row = inflater.inflate(R.layout.*activity\_list*, parent, false);  
  
 TextView title = (TextView) row.findViewById(R.id.*filmTitle*); *// связь данных и ИД слоя* TextView year = (TextView) row.findViewById(R.id.*filmReleasedDetail*);  
 TextView type = (TextView) row.findViewById(R.id.*filmType*);  
  
 title.setText(handle(filmsToShow.get(position).getTitle())); *// запись всех параметров* year.setText("Year: " + handle(filmsToShow.get(position).getYear()));  
 type.setText("Type: " + handle(filmsToShow.get(position).getType())+" "+freeSpace);  
  
 ImageView iconImageView = (ImageView) row.findViewById(R.id.*poster*);  
  
 String posterName = filmsToShow.get(position).getPoster();  
 int res = getContext().getResources().getIdentifier(posterName.replaceAll(".jpg",  
 ""), "drawable", getContext().getPackageName()); *// поиск ИД по имени* if(res!=0) iconImageView.setImageResource(res); *// если нет такого ИД* else {  
 try { *// пробуем установить пользовательское изображение* File imageFile = new File(getContext().getFilesDir() + "/" + posterName); *// пользовательское изображение* InputStream is = new FileInputStream(imageFile);  
  
 Bitmap userImage = BitmapFactory.*decodeStream*(is); *// фото в стрим* iconImageView.setImageBitmap(userImage); *// установка фото* } catch (Exception e) {iconImageView.setImageResource(R.drawable.*kpi\_logo*);} *// стандартная картинка* }  
 return row;  
 }  
  
 public String handle(String str){ *// обработчик строки* if(str.equals("")) return "None"; *// если не задан любой из параметров* else return str;  
  
 }  
 }  
}

**Film.java**

package ua.kpi.comsys.IO8206;  
  
public class Film {  
 private String Title, Year, Type, imdbID, Poster, Rated, Runtime, Genre, imdbRating, imdbVotes, Released, Production,  
 Language, Country, Awards, Director, Writer, Actors, Plot;  
  
 public Film(String title, String year, String type, String imdbID, String poster, String rated, String production,  
 String runtime, String genre, String imdbRating, String imdbVotes, String language,  
 String country, String awards, String director, String writer, String actors, String plot, String released) {  
 this.Title = title;  
 this.Year = year;  
 this.Type = type;  
 this.imdbID = imdbID;  
 this.Poster = poster;  
 this.Rated = rated;  
 this.Runtime = runtime;  
 this.Genre = genre;  
 this.imdbRating = imdbRating;  
 this.imdbVotes = imdbVotes;  
 this.Language = language;  
 this.Country = country;  
 this.Awards = awards;  
 this.Director = director;  
 this.Writer = writer;  
 this.Actors = actors;  
 this.Plot = plot;  
 this.Released = released;  
 this.Production = production;  
 }  
  
 public String getProduction() {  
 return Production;  
 }  
  
 public void setProduction(String production) {  
 Production = production;  
 }  
  
 public String getReleased() {  
 return Released;  
 }  
  
 public void setReleased(String released) {  
 Released = released;  
 }  
  
 public String getTitle() {  
 return Title;  
 }  
  
 public void setTitle(String title) {  
 this.Title = title;  
 }  
  
 public String getYear() {  
 return Year;  
 }  
  
 public void setYear(String year) {  
 this.Year = year;  
 }  
  
 public String getType() {  
 return Type;  
 }  
  
 public void setType(String type) {  
 this.Type = type;  
 }  
  
 public String getImdbID() {  
 return imdbID;  
 }  
  
 public void setImdbID(String imdbID) {  
 this.imdbID = imdbID;  
 }  
  
 public String getPoster() {  
 return Poster;  
 }  
  
 public void setPoster(String poster) {  
 this.Poster = poster;  
 }  
  
 public String getRated() {  
 return Rated;  
 }  
  
 public void setRated(String rated) {  
 Rated = rated;  
 }  
  
 public String getRuntime() {  
 return Runtime;  
 }  
  
 public void setRuntime(String runtime) {  
 Runtime = runtime;  
 }  
  
 public String getGenre() {  
 return Genre;  
 }  
  
 public void setGenre(String genre) {  
 Genre = genre;  
 }  
  
 public String getImdbRating() {  
 return imdbRating;  
 }  
  
 public void setImdbRating(String imdbRating) {  
 this.imdbRating = imdbRating;  
 }  
  
 public String getImdbVotes() {  
 return imdbVotes;  
 }  
  
 public void setImdbVotes(String imdbVotes) {  
 this.imdbVotes = imdbVotes;  
 }  
  
 public String getLanguage() {  
 return Language;  
 }  
  
 public void setLanguage(String language) {  
 Language = language;  
 }  
  
 public String getCountry() {  
 return Country;  
 }  
  
 public void setCountry(String country) {  
 Country = country;  
 }  
  
 public String getAwards() {  
 return Awards;  
 }  
  
 public void setAwards(String awards) {  
 Awards = awards;  
 }  
  
 public String getDirector() {  
 return Director;  
 }  
  
 public void setDirector(String director) {  
 Director = director;  
 }  
  
 public String getWriter() {  
 return Writer;  
 }  
  
 public void setWriter(String writer) {  
 Writer = writer;  
 }  
  
 public String getActors() {  
 return Actors;  
 }  
  
 public void setActors(String actors) {  
 Actors = actors;  
 }  
  
 public String getPlot() {  
 return Plot;  
 }  
  
 public void setPlot(String plot) {  
 Plot = plot;  
 }  
  
 @Override  
 public String toString() {  
*// return "Films{" +  
// "title='" + Title + '\'' +  
// ", year='" + Year + '\'' +  
// ", contentType='" + Type + '\'' +  
// ", imdb='" + imdbID + '\'' +  
// ", poster='" + Poster + '\'' +  
// '}';* return Title + '\n' +  
 Year + '\n' +  
 "Type: " + Type;  
 }  
}

**JsonHelper.java**

package ua.kpi.comsys.IO8206;  
  
import android.content.Context;  
import android.content.res.Resources;  
  
import com.google.gson.Gson;  
  
import java.io.ByteArrayOutputStream;  
import java.io.File;  
import java.io.FileInputStream;  
import java.io.FileNotFoundException;  
import java.io.FileOutputStream;  
import java.io.FileWriter;  
import java.io.IOException;  
import java.io.InputStream;  
import java.io.InputStreamReader;  
import java.util.List;  
  
public class JsonHelper {  
 private static int *RESOURCE\_NAME*;  
 private static String *FILE\_USER\_NAME*;  
 private static Boolean *userFileEnable* = false;  
 private static File *f*;  
  
 public JsonHelper(int resourceName){  
 *RESOURCE\_NAME* = resourceName;  
 }  
  
 public static void setFileUserName(String fileUserName) {  
 *FILE\_USER\_NAME* = fileUserName;  
 }  
  
 public static void setUserFileEnable(Boolean userFileEnable) {  
 JsonHelper.*userFileEnable* = userFileEnable;  
 }  
  
 public static boolean exportToJSON(Context context, List<Film> dataList) { *// запись в файл* Gson gson = new Gson();  
 DataItems dataItems = new DataItems();  
 dataItems.setSearch(dataList);  
 String jsonString = gson.toJson(dataItems);  
  
 FileOutputStream fileOutputStream = null;  
  
 try {  
 fileOutputStream = context.openFileOutput(*FILE\_USER\_NAME*, Context.*MODE\_PRIVATE*);  
 fileOutputStream.write(jsonString.getBytes());  
 *userFileEnable* = true;  
 return true;  
 } catch (Exception e) {  
 e.printStackTrace();  
 } finally {  
 if (fileOutputStream != null) {  
 try {  
 fileOutputStream.close();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
 }  
 return false;  
 }  
  
  
 public static List<Film> importFilmListFromJSON(Context context) {  
 InputStreamReader streamReader = null;  
 FileInputStream fileInputStream = null;  
  
 try{  
 Gson gson = new Gson();  
  
 *f* = new File(context.getFilesDir() + "/"+*FILE\_USER\_NAME*);  
 if(*f*.exists()){ *// файл найден  
 userFileEnable* = true;  
 }  
 else{*// файл не найден* try(FileWriter writer = new FileWriter(*f*)){  
 writer.write(*getStringFromRawFile*(context)); *// запись в файл юзерспейса JSON`а* writer.flush();  
 *userFileEnable* = true;  
 }  
 catch(IOException ex){  
 ex.printStackTrace();  
 }  
 }  
  
 DataItems dataItems = gson.fromJson(*getStringFromRawFile*(context), DataItems.class); *// создание объектов из файла* return dataItems.getSearch();  
 }  
 catch (Exception ex){  
 ex.printStackTrace();  
 }  
 finally {  
 if (streamReader != null) {  
 try {  
 streamReader.close();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
 if (fileInputStream != null) {  
 try {  
 fileInputStream.close();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
 }  
  
  
 return null;  
 }  
  
 public static Film importFilmFromJSON(Context context) {  
 InputStreamReader streamReader = null;  
 FileInputStream fileInputStream = null;  
  
 try{  
 Gson gson = new Gson();  
 Film film = gson.fromJson(*getStringFromRawFile*(context), Film.class); *// создание объекта из файла* return film;  
 }  
 catch (Exception ex){  
 ex.printStackTrace();  
 }  
 finally {  
 if (streamReader != null) {  
 try {  
 streamReader.close();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
 if (fileInputStream != null) {  
 try {  
 fileInputStream.close();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
 }  
  
 return null;  
 }  
  
 private static class DataItems {  
 private List<Film> Search;  
  
 List<Film> getSearch() {  
 return Search;  
 }  
 void setSearch(List<Film> search) {  
 this.Search = search;  
 }  
 }  
  
*// private static class DataItem {  
// public Film Title;  
//  
// Film getTitle() {  
// return Title;  
// }  
// void setTitle(Film title) {  
// this.Title = title;  
// }  
// }* public static String getStringFromRawFile(Context context) {  
 InputStream is = null;  
 if(!*userFileEnable*) { *// если НЕ доступна пользовательская версия списка фильмов* Resources r = context.getResources();  
 is = r.openRawResource(*RESOURCE\_NAME*);  
 }  
 else { *// если доступна пользовательская версия списка фильмов* try {  
 is = new FileInputStream(*f*);  
 } catch (FileNotFoundException e) {  
 e.printStackTrace();  
 }  
 }  
  
 String myText = null;  
 try {  
 myText = *convertStreamToString*(is);  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 try {  
 is.close();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 return myText;  
 }  
  
 static String convertStreamToString(InputStream is) throws IOException {  
 ByteArrayOutputStream baos = new ByteArrayOutputStream();  
 int i = is.read();  
 while( i != -1)  
 {  
 baos.write(i);  
 i = is.read();  
 }  
 return baos.toString();  
 }  
}

**FilmDetail.java**

package ua.kpi.comsys.IO8206.ui;  
  
import android.os.Bundle;  
import android.text.method.ScrollingMovementMethod;  
import android.view.LayoutInflater;  
import android.view.View;  
import android.widget.ImageView;  
import android.widget.TextView;  
import android.widget.Toast;  
  
import androidx.annotation.Nullable;  
import androidx.appcompat.app.AppCompatActivity;  
  
import ua.kpi.comsys.IO8206.Film;  
import ua.kpi.comsys.IO8206.JsonHelper;  
import ua.kpi.comsys.IO8206.R;  
  
public class FilmDetail extends AppCompatActivity {  
 Film film;  
 String imbdId;  
  
 @Override  
 protected void onCreate(@Nullable Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*film\_info*);  
  
 Bundle arguments = getIntent().getExtras();  
 imbdId = arguments.get("filmImdbId").toString();  
  
 int res = this.getResources().getIdentifier(imbdId, "raw", this.getPackageName()); *// поиск ИД по имени* JsonHelper jsonHelper = new JsonHelper(res);  
 jsonHelper.*setUserFileEnable*(false);  
  
  
 if(res!=0) {  
 film = jsonHelper.*importFilmFromJSON*(this); *// если есть такой ИД* TextView title = findViewById(R.id.*filmTitleDetail*);  
 TextView releasedDate = findViewById(R.id.*filmReleasedDetail*);  
 TextView rated = findViewById(R.id.*filmRatedDetail*);  
 TextView runtime = findViewById(R.id.*filmRuntimeDetail*);  
 TextView genre = findViewById(R.id.*filmGenreDetail*);  
 TextView imdbRating = findViewById(R.id.*filmImdbRatingDetail*);  
 TextView imdbVotes = findViewById(R.id.*filmImdbVotesDetail*);  
 TextView production = findViewById(R.id.*filmProductionDetail*);  
 TextView type = findViewById(R.id.*filmTypeDetail*);  
 TextView director = findViewById(R.id.*filmDirectorDetail*);  
 TextView writer = findViewById(R.id.*filmWriterDetail*);  
 TextView language = findViewById(R.id.*filmLanguageDetail*);  
 TextView country = findViewById(R.id.*filmCountryDetail*);  
 TextView awards = findViewById(R.id.*filmAwardsDetail*);  
 TextView actors = findViewById(R.id.*filmActorsDetail*);  
 TextView plot = findViewById(R.id.*filmPlotDetail*);  
  
 ImageView poster = (ImageView) findViewById(R.id.*filmPosterDetail*);  
  
 int img;  
 try {  
 String posterName = film.getPoster().replaceAll(".jpg","").toLowerCase();  
 img = getResources().getIdentifier(posterName, "drawable", getPackageName()); *// поиск ИД по имени* } catch (Exception e){img = 0;};  
  
  
 if(img!=0) poster.setImageResource(img); *// если есть такой ИД* else poster.setImageResource(R.drawable.*kpi\_logo*); *// стандартная картинка* title.setText(film.getTitle());  
 releasedDate.setText("Released: "+film.getReleased());  
 rated.setText("Rated: "+film.getRated());  
 runtime.setText("Runtime: "+film.getRuntime());  
 genre.setText("Genre: "+film.getGenre());  
 imdbRating.setText("IMDB: "+film.getImdbRating());  
 imdbVotes.setText("("+film.getImdbVotes()+")");  
 production.setText("Production: "+film.getProduction());  
 type.setText("Type: "+film.getType());  
 director.setText("Director: "+film.getDirector());  
 writer.setText("Writer: "+film.getWriter());  
 language.setText("Language: "+film.getLanguage());  
 country.setText("Country: "+film.getCountry());  
 awards.setText("Awards: "+film.getAwards());  
 actors.setText("Actors: "+film.getActors());  
 plot.setText(film.getPlot());  
  
 }  
 else {Toast.*makeText*(this, "Information not found", Toast.*LENGTH\_LONG*).show(); finish();}  
 }  
  
 public void setFilm(Film film) {  
 this.film = film;  
 }  
}

**AddFilmActivity.java**

package ua.kpi.comsys.IO8206;  
  
import androidx.appcompat.app.AppCompatActivity;  
import androidx.fragment.app.Fragment;  
import androidx.fragment.app.FragmentTransaction;  
  
import android.content.Context;  
import android.content.Intent;  
import android.graphics.Bitmap;  
import android.graphics.BitmapFactory;  
import android.graphics.drawable.BitmapDrawable;  
import android.net.Uri;  
import android.os.Bundle;  
import android.view.View;  
import android.widget.Button;  
import android.widget.EditText;  
import android.widget.ImageView;  
import android.widget.Toast;  
  
import java.io.ByteArrayOutputStream;  
import java.io.File;  
import java.io.FileNotFoundException;  
import java.io.FileOutputStream;  
import java.io.IOException;  
import java.io.InputStream;  
import java.io.OutputStream;  
import java.util.List;  
import java.util.Random;  
  
import ua.kpi.comsys.IO8206.ui.films.FilmsList;  
  
public class AddFilmActivity extends AppCompatActivity {  
 int resFilmList;  
 private ImageView imageView;  
 private final int Pick\_image = 1;  
 Button choosePoster;  
 Bitmap selectedImage;  
 private List<Film> films;  
 OutputStream os;  
 ByteArrayOutputStream bos;  
 Random random = new Random();  
 Boolean uploadedImage = false;  
 String userFileMovie = "movieslistuser.txt"; *// стандартное значение, которое заменится* @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
  
*// try {  
// userFileMovie = getResources().getString(R.string.user\_films\_list); // берём имя файла из ресурсов  
// } catch (Exception e){}* setContentView(R.layout.*activity\_add\_film*);  
  
 imageView = (ImageView) findViewById(R.id.*imageView\_filmAdd*);  
 choosePoster = (Button) findViewById(R.id.*buttonChoosePoster\_filmAdd*);  
 bos = new ByteArrayOutputStream();  
  
 Bundle arguments = getIntent().getExtras();  
 resFilmList = (int)arguments.get("moviesListId");  
 }  
  
 public void choosePoster(View view) { *// кнопка выбрать изображение* Intent photoPickerIntent = new Intent(Intent.*ACTION\_PICK*);  
 photoPickerIntent.setType("image/\*"); *// получаемый тип* startActivityForResult(photoPickerIntent, Pick\_image); *// ожидание выбора фото* uploadedImage = true;  
 }  
  
 public void addBtn(View view) { *// кнопка добавить* String filmTitle = ((EditText)findViewById(R.id.*editTextFilmTitle\_filmAdd*)).getText().toString();  
 String filmYear = ((EditText)findViewById(R.id.*editTextFilmYear\_filmAdd*)).getText().toString();  
 String filmType = ((EditText)findViewById(R.id.*editTextFilmType\_filmAdd*)).getText().toString();  
  
 String newPosterName;  
  
 if(filmTitle.length()<1){ *// некорректный заголовок* Toast.*makeText*(view.getContext(), "Uncorrected title", Toast.*LENGTH\_LONG*).show();  
 }  
 else {  
 JsonHelper jsonHelper = new JsonHelper(resFilmList);  
 jsonHelper.*setFileUserName*(userFileMovie);  
 films = jsonHelper.*importFilmListFromJSON*(view.getContext()); *// берём фильмы из файла* if(uploadedImage){ *// если выбрано изображение* newPosterName = "poster\_"+(random.nextInt(99999)+100) + ".png";  
  
 selectedImage.compress(Bitmap.CompressFormat.*PNG*, 100, bos);  
  
 byte[] bitmapdata = bos.toByteArray();  
  
  
*// File filesDir = view.getContext().getFilesDir();* File imageFile = new File(view.getContext().getFilesDir(), newPosterName);  
  
 try {  
 FileOutputStream fos = new FileOutputStream(imageFile);  
 fos.write(bitmapdata);  
 fos.flush();  
 fos.close();  
 } catch (FileNotFoundException e) {  
 e.printStackTrace();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
  
  
 }  
 else newPosterName = "";  
  
 films.add(new Film(filmTitle, filmYear, filmType, "", newPosterName, "",  
 "", "", "", "", "", "",  
 "", "", "", "", "", "", "" ));  
  
 jsonHelper.*exportToJSON*(view.getContext(), films);  
  
 finish();  
 Toast.*makeText*(view.getContext(), "Added successfully", Toast.*LENGTH\_LONG*).show();  
 }  
 }  
  
 public void returnBtn(View view) { *// кнопка возврата* finish();  
 }  
  
 @Override  
 protected void onActivityResult(int requestCode, int resultCode, Intent imageReturnedIntent) { *// загрузка фото* super.onActivityResult(requestCode, resultCode, imageReturnedIntent);  
  
 if (requestCode == Pick\_image) {  
 if (resultCode == *RESULT\_OK*) {  
 try {  
 final Uri imageUri = imageReturnedIntent.getData(); *// получить URI изображения* final InputStream imageStream = getContentResolver().openInputStream(imageUri); *// получить стрим* selectedImage = BitmapFactory.*decodeStream*(imageStream); *// преобразовать в битмап* imageView.setImageBitmap(selectedImage); *// показать в imageView* choosePoster.setBackgroundColor(getResources().getColor(R.color.*posterChooseSuccess*)); *// фон кнопки* } catch (FileNotFoundException e) {  
 choosePoster.setBackgroundColor(getResources().getColor(R.color.*posterChooseFailure*));  
 e.printStackTrace();  
 }  
 }  
 }  
 }  
}

**Висновоки:**

Виконано лабораторку роботу №4 за поставленою задачею. Реалізовано пошук по елементам списку, додавання нових елементів з фото та видалення елементів.

Додаток коректно працює.