НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО”

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

Лабораторна робота №6

з дисципліни

“Програмування мобільних систем”

Виконав:

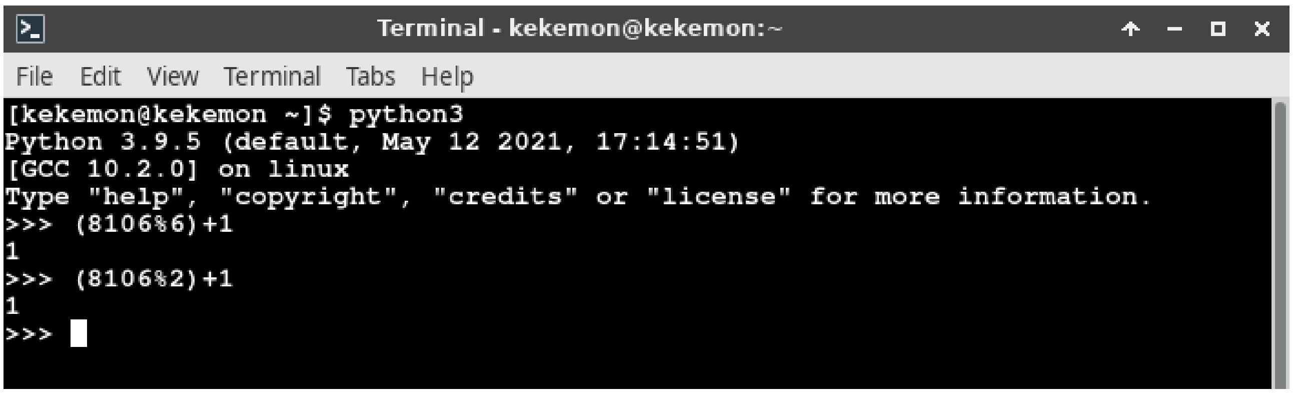
студент групи ІВ-81

ЗК 8106

Бухтій О.В.

Київ 2021

1. Визначення варіанту:

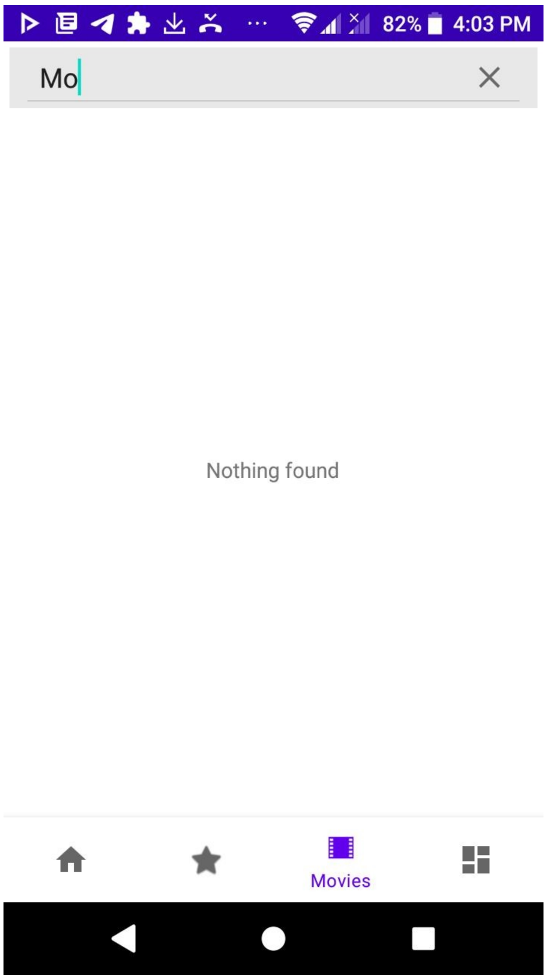

Terminal - kekemon@kekemon:~
File Edit View Terminal Tabs Help

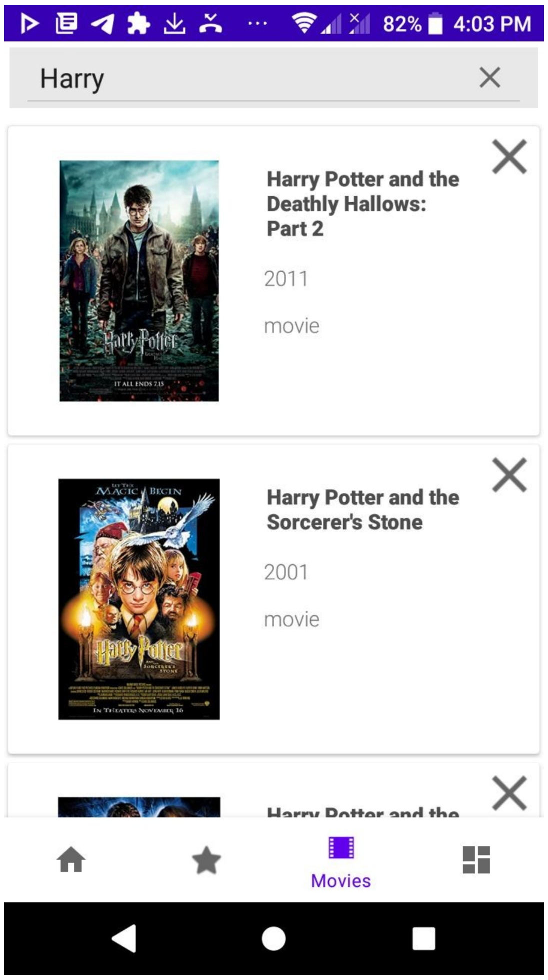
```
[kekemon@kekemon ~]$ python3
Python 3.9.5 (default, May 12 2021, 17:14:51)
[GCC 10.2.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> (8106%6)+1
1
>>> (8106%2)+1
1
>>>
```

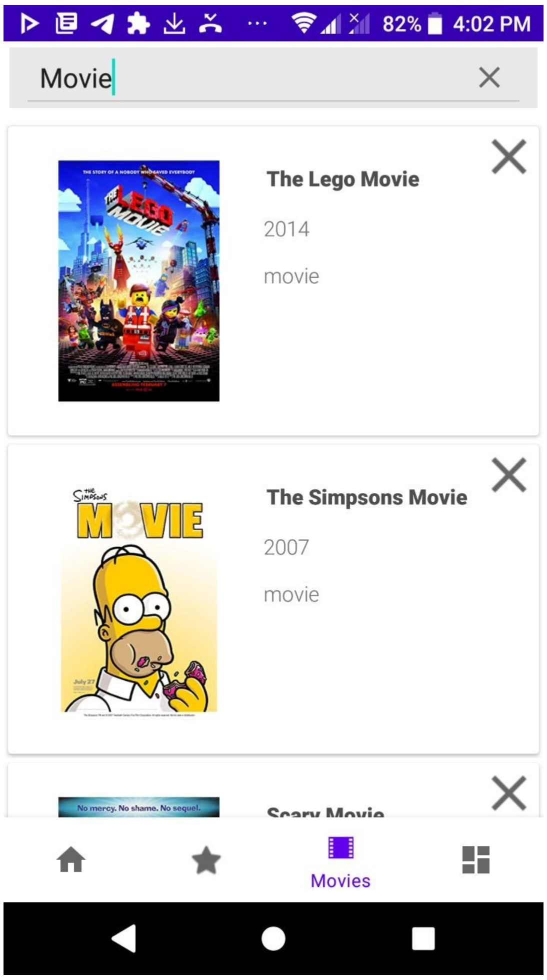
Посилання на репозиторій: https://github.com/KekemonBS/PMS/tree/main/LAB_7

2. Виконання:

Скріншоти:




82% 4:03 PM
Harry
Harry Potter and the Deathly Hallows: Part 2
2011
movie
Harry Potter and the Sorcerer's Stone
2001
movie
Harry Potter and the
Movies


82% 4:02 PM

Movie

The Lego Movie

2014

movie

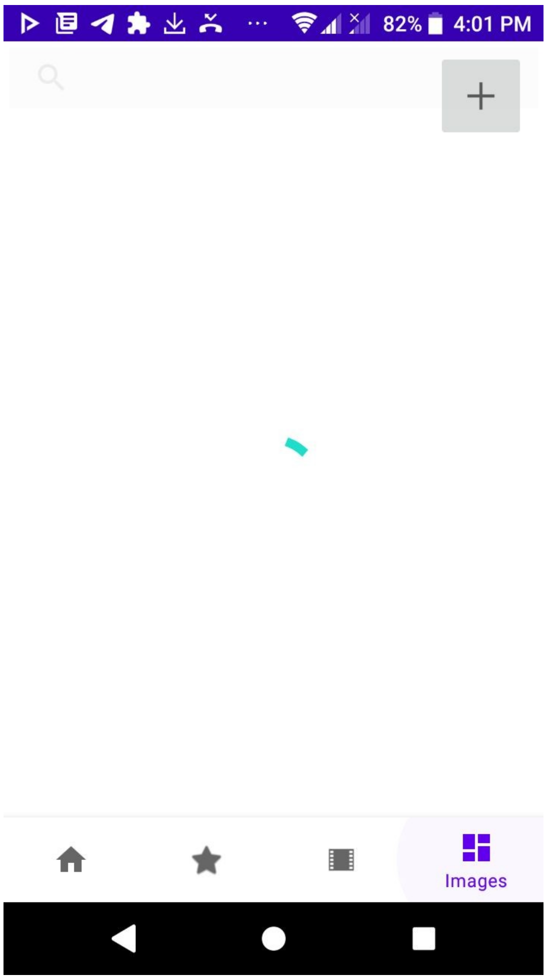
The Simpsons Movie

2007

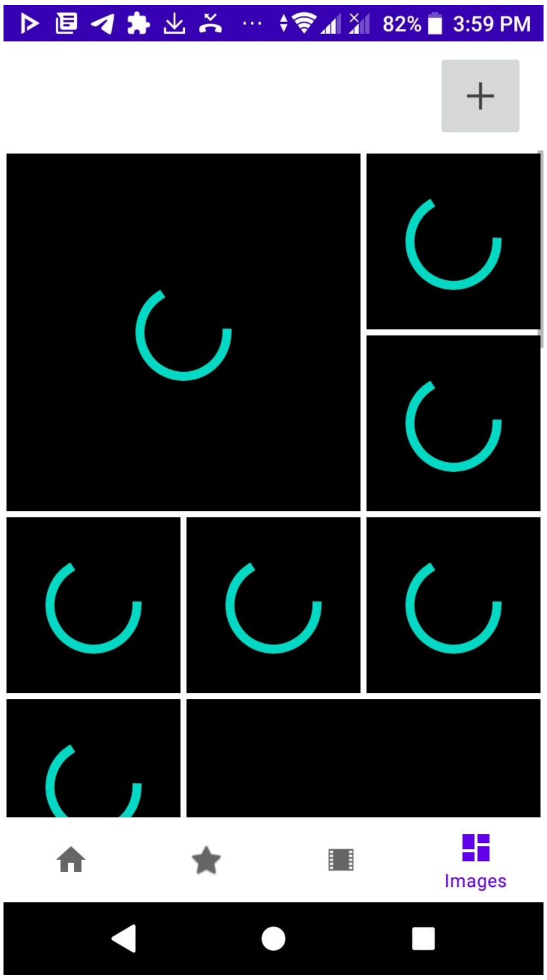
movie

Scary Movie

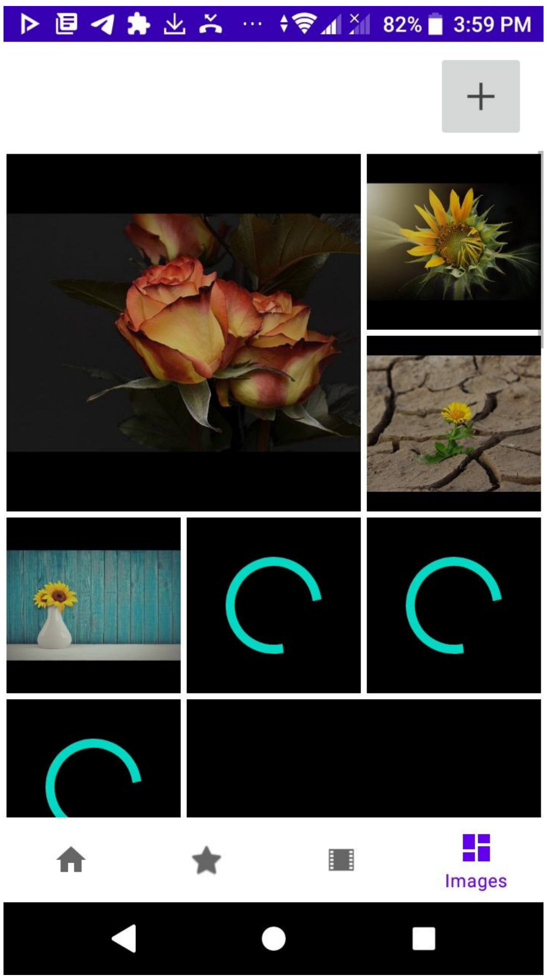
Movies


82% 4:01 PM

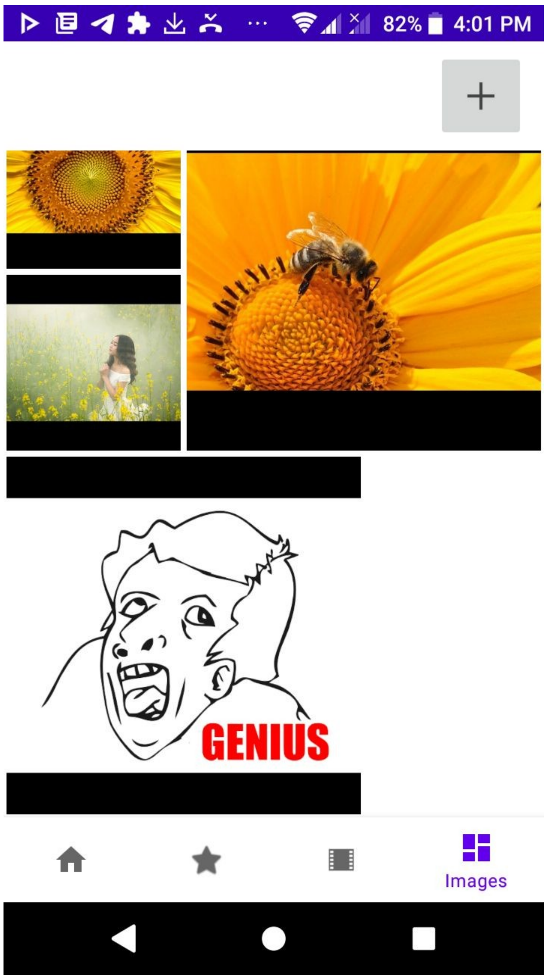
Images


82% 3:59 PM

Images


82% 3:59 PM

Images



3. **Лістинг коду (основні що були створені/ змінені):**

```
---GalleryAdapter.java---
package ua.kpi.comsys.iv8106.adapters;

import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ImageView;
import android.widget.ProgressBar;

import androidx.annotation.NonNull;
import androidx.fragment.app.Fragment;
import androidx.recyclerview.widget.RecyclerView;

import com.squareup.picasso.Callback;
import com.squareup.picasso.Picasso;

import java.util.ArrayList;

import ua.kpi.comsys.iv8106.R;
import ua.kpi.comsys.iv8106.model.ImageItem;
```

```
public class GalleryAdapter extends
RecyclerView.Adapter<GalleryAdapter.GalleryViewHolder> {

    private final Fragment             fragment;
    private final ArrayList<ImageItem> images;

    public GalleryAdapter(Fragment fragment, ArrayList<ImageItem>
images) {
        this.fragment = fragment;
        this.images = images;
    }

    public class GalleryViewHolder extends RecyclerView.ViewHolder
{

        private ImageView iw;
        private ProgressBar spinnerImg;

        public GalleryViewHolder(@NonNull View itemView) {
            super(itemView);
            this.iw = (ImageView)
itemView.findViewById(R.id.image);
            this.spinnerImg =
(ProgressBar)itemView.findViewById(R.id.progressBarImg);

        }
    }

    @NonNull
    @Override
    public GalleryAdapter.GalleryViewHolder
onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
        LayoutInflater inflater =
LayoutInflater.from(parent.getContext());
        View view = inflater.inflate(R.layout.image, parent,
false);
        return new GalleryAdapter.GalleryViewHolder(view);
    }
//------------------------------------------------------------------
---------------------------------------------
    @Override
    public void onBindViewHolder(@NonNull
GalleryAdapter.GalleryViewHolder holder, int position) {
        //holder.itemView.getLayoutParams().height = 650;
        //holder.iw.setImageURI(images.get(position));

        holder.spinnerImg.setVisibility(ProgressBar.VISIBLE);
        if (images.get(position).getBitmap() == null) {
            Picasso.get()

.load(images.get(position).getWebformatURL())/*.placeholder()*/
                    .into(holder.iw, new Callback() {
                        @Override
                        public void onSuccess() {

holder.spinnerImg.setVisibility(ProgressBar.INVISIBLE);
                        }

                        @Override
                        public void onError(Exception e) {

                        }
                    });
        } else {

holder.iw.setImageBitmap(images.get(position).getBitmap());

holder.spinnerImg.setVisibility(ProgressBar.INVISIBLE);
        }

    }
//-------------------------------------------------------------------
---------------------------------------------
    @Override
    public int getItemCount() {
        return images.size();
    }

}
```

---MoviesAdapter.java---

```
package ua.kpi.comsys.iv8106.adapters;

import android.content.Intent;
import android.view.LayoutInflater;
import android.view.MotionEvent;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ImageView;
import android.widget.TextView;
import android.widget.Toast;

import androidx.annotation.NonNull;
import androidx.fragment.app.Fragment;
import androidx.recyclerview.widget.RecyclerView;
import com.squareup.picasso.Picasso;

import java.util.ArrayList;

import ua.kpi.comsys.iv8106.R;
import ua.kpi.comsys.iv8106.model.MovieItem;
import
ua.kpi.comsys.iv8106.secondary_activities.MovieDetailsActivity;

public class MoviesAdapter extends
RecyclerView.Adapter<MoviesAdapter.MovieViewHolder> {
    private final Fragment context;
    private final ArrayList<MovieItem> movies;
    private final ArrayList<String> maintitle;

    public MoviesAdapter(Fragment context, ArrayList<MovieItem>
movies, ArrayList<String> maintitle) {
        this.context=context;
        this.movies=movies;
        this.maintitle = maintitle;
    }

    public class MovieViewHolder extends RecyclerView.ViewHolder {

        private ImageView image;
        private TextView titleText;
        private TextView yearText;
        private TextView typeText;

        private ImageView deleteButton;

        public MovieViewHolder(View view) {
            super(view);
            // Define click listener for the ViewHolder's View

            this.image     = (ImageView)
view.findViewById(R.id.image);
            this.titleText = (TextView)
view.findViewById(R.id.title);
            this.yearText  = (TextView)
view.findViewById(R.id.year);
            this.typeText  = (TextView)
view.findViewById(R.id.type);

            this.deleteButton = (ImageView)
view.findViewById(R.id.deleteButton);

        }
    }

    // Create new views (invoked by the layout manager)
    @Override
    public MovieViewHolder onCreateViewHolder(ViewGroup parent,
int viewType) {
        // Create a new view, which defines the UI of the list
item
        LayoutInflater inflater =
LayoutInflater.from(parent.getContext());
        View view = inflater.inflate(R.layout.movie, parent,
false);

        return new MovieViewHolder(view);
    }



    @Override
    public void onBindViewHolder(@NonNull MovieViewHolder holder,
int position) {
//        int drawableResourceId =
context.getResources().getIdentifier(
//
movies.get(position).getPoster().toLowerCase().replace(".jpg",
""),
//                "drawable",
context.getContext().getPackageName());

Picasso.get().load(movies.get(position).getPoster()).into(holder.i
mage);

        if (!movies.get(position).isVisible()) {
            holder.itemView.setVisibility(View.INVISIBLE);
            holder.itemView.getLayoutParams().height = 0;
        } else {
            holder.itemView.setVisibility(View.VISIBLE);
            //holder.itemView.getLayoutParams().height = 650;
        }

        holder.titleText.setText(maintitle.get(position));
        holder.yearText.setText(movies.get(position).getYear());
        holder.typeText.setText(movies.get(position).getType());

//        if (drawableResourceId != 0) {
//            holder.image.setImageResource(drawableResourceId);
//        } else {
```

```
//
holder.image.setImageResource(R.drawable.ic_action_cancel);
//        }

        holder.deleteButton.setOnTouchListener(new
View.OnTouchListener() {
            @Override
            public boolean onTouch(View v, MotionEvent event) {
                removeItem(holder.getAdapterPosition());
                return true;
            }
        });

        holder.itemView.setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent = new Intent(context.getContext(),
MovieDetailsActivity.class);
                if ((!
movies.get(position).getImdbId().equals(null)) &&
                    (!
movies.get(position).getImdbId().equals("noid"))) {
                    intent.putExtra("id",
movies.get(position).getImdbId());
                    context.startActivity(intent);
                } else {

System.out.println(movies.get(position).getImdbId());
                    Toast toast =
Toast.makeText(context.getContext(), "No ID", Toast.LENGTH_SHORT);
                    toast.show();
                }
            }
        });
    }

    @Override
    public int getItemCount() {
        return movies.size();
    }

    public void removeItem(int position) {
        if (position == -1)
            return;
        this.movies.remove(position);
        this.maintitle.remove(position);
        notifyDataSetChanged();
    }
}
```

---GalleryFragment.java---

```
package ua.kpi.comsys.iv8106.ui.gallery;

import android.app.Activity;
import android.content.Intent;
import android.graphics.Bitmap;
import android.net.Uri;
import android.os.Bundle;
import android.provider.MediaStore;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ImageButton;
import android.widget.ProgressBar;

import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.fragment.app.Fragment;
import androidx.recyclerview.widget.RecyclerView;

import com.arasthel.spannedgridlayoutmanager.SpanSize;
import
com.arasthel.spannedgridlayoutmanager.SpannedGridLayoutManager;
import com.google.gson.Gson;
import com.google.gson.JsonObject;
import com.google.gson.reflect.TypeToken;

import java.io.IOException;
import java.lang.reflect.Type;
import java.util.ArrayList;
import java.util.LinkedList;
import java.util.Queue;

import kotlin.jvm.functions.Function1;
import ua.kpi.comsys.iv8106.R;
import ua.kpi.comsys.iv8106.adapters.GalleryAdapter;
import ua.kpi.comsys.iv8106.model.ImageItem;
import ua.kpi.comsys.iv8106.requester.Requester;

public class GalleryFragment extends Fragment {

    private int RESULT_LOAD_IMG = 1;
    private static String response;
    //public ArrayList<Bitmap> images = new ArrayList<>();
    Type listOfImagesItemsType = new
TypeToken<ArrayList<ImageItem>>() {}.getType();
    ArrayList<ImageItem> images   = new ArrayList<>();



    public GalleryAdapter adapter_gallery = new
GalleryAdapter(this, images);
    @Nullable
    @Override
    public View onCreateView(@NonNull LayoutInflater inflater,
                             @Nullable ViewGroup container,
                             @Nullable Bundle savedInstanceState)
{

        this.setRetainInstance(true);

        View view = inflater.inflate(R.layout.fragment_gallery,
container, false);
        RecyclerView recycle =
view.findViewById(R.id.galleryRecyclerView);
        recycle.setNestedScrollingEnabled(false);

        SpannedGridLayoutManager spannedGridLayoutManager = new
SpannedGridLayoutManager(
              SpannedGridLayoutManager.Orientation.VERTICAL, 3);
        spannedGridLayoutManager.setItemOrderIsStable(false);

        spannedGridLayoutManager.setSpanSizeLookup(new
SpannedGridLayoutManager.SpanSizeLookup(new Function1<Integer,
SpanSize>(){
            @Override public SpanSize invoke(Integer position) {
                if (position % 9 == 0) {
                    return new SpanSize(2, 2);
                } else if ((position - 7)  % 9 == 0) {
                    return new SpanSize(2, 2);
                } else {
                    return new SpanSize(1, 1);
                }
            }
        }));

        ImageButton addButton =
view.findViewById(R.id.moreImageButton);
        ProgressBar spinner =
(ProgressBar)view.findViewById(R.id.progressBar);
        spinner.setVisibility(ProgressBar.VISIBLE);
        recycle.setLayoutManager(spannedGridLayoutManager);
        recycle.setAdapter(adapter_gallery);


        //------------------------------------------------------
-------------------------
        //Perform request in separate thread
        new Thread(new Runnable() {
            @Override
            public void run() {
                String formattedUrlString =
"https://pixabay.com/api/?key=%s&q=%s&image_type=photo&per_page=
%s";
                String apiKey = "19193969-
87191e5db266905fe8936d565";
                String request = "yellow+flowers";
                String count = "27";
                Queue<String> queue = new LinkedList<>();
                Requester req = new Requester(queue,
formattedUrlString, apiKey, request, count);
                Thread th1 = new Thread(req, "images");
                th1.start();
                try {
                    th1.join();
                    setJSONResponse(queue.remove());
                } catch (InterruptedException e) {
                    e.printStackTrace();
                }
                System.out.println(response);

                //Update data (view was already retrieved by now
for shure)
                Gson gson = new Gson();
                if (response != null &&
                        gson.fromJson(response,
JsonObject.class).has("hits")) {
                    spinner.setVisibility(ProgressBar.INVISIBLE);
                    JsonObject gsontmp = gson.fromJson(response,
JsonObject.class);
                    images.clear();

images.addAll(gson.fromJson(gsontmp.get("hits"),
listOfImagesItemsType));

                } else {
                    images.clear();
                    images.clear();
                }
                //Notify that data changed
                getActivity().runOnUiThread(new Runnable() {
                    @Override
                    public void run() {
                        adapter_gallery.notifyDataSetChanged();
                    }
                });

                spinner.setVisibility(ProgressBar.INVISIBLE);
            }
        }).start();
        //------------------------------------------------------------
--------------------------

        addButton.setOnClickListener(new View.OnClickListener() {
            public void onClick(View v) {
                Intent photoPickerIntent = new
Intent(Intent.ACTION_PICK);
                photoPickerIntent.setType("image/*");
                startActivityForResult(photoPickerIntent,
RESULT_LOAD_IMG);
            }
        });

        System.out.println("HERE1");

        return view;
    }

    public static void setJSONResponse(String JSON) {
        response = JSON;
    }



    @Override
    public void onActivityResult(int requestCode, int resultCode,
Intent picker) {
        super.onActivityResult(requestCode, resultCode, picker);
        if (requestCode == 1) {
            if (resultCode == Activity.RESULT_OK) {
                Uri uri = picker.getData();
                Bitmap selectedImage = null;
                try {
                    selectedImage =
MediaStore.Images.Media.getBitmap(
                            getContext().getContentResolver(),
uri);
                } catch (IOException e) {
                    e.printStackTrace();
                }
                Bitmap scaled =
selectedImage.createScaledBitmap(selectedImage,

(int)Math.ceil(selectedImage.getWidth()/2),
(int)Math.ceil(selectedImage.getHeight()/2), false);
                ImageItem selectedimageItem = new ImageItem();
                selectedimageItem.setBitmap(scaled);
                images.add(selectedimageItem);
                adapter_gallery.notifyDataSetChanged();
            }
        }
    }
}
```

---MoviesFragment.java---

```
package ua.kpi.comsys.iv8106.ui.movies;

import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ProgressBar;
import android.widget.SearchView;
import android.widget.TextView;

import androidx.annotation.NonNull;
import androidx.fragment.app.Fragment;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;

import com.google.gson.Gson;
import com.google.gson.JsonObject;
import com.google.gson.reflect.TypeToken;

import java.lang.reflect.Type;
import java.util.ArrayList;
import java.util.LinkedList;
import java.util.Queue;

import ua.kpi.comsys.iv8106.R;
import ua.kpi.comsys.iv8106.adapters.MoviesAdapter;
import ua.kpi.comsys.iv8106.model.MovieItem;
import ua.kpi.comsys.iv8106.requester.Requester;
```

```
public class MoviesFragment extends Fragment {

    private static String response;
    //    private static final int REQUEST_READ_EXTERNAL_STORAGE =
1;
    View root;

    Type listOfMoviesItemsType = new
TypeToken<ArrayList<MovieItem>>() {}.getType();
    ArrayList<MovieItem> movie_list = new ArrayList<>();
    ArrayList<String> main_title = new ArrayList<>();


    public View onCreateView(@NonNull LayoutInflater inflater,
                             ViewGroup container, Bundle
savedInstanceState) {
        this.setRetainInstance(true);

        root = inflater.inflate(R.layout.fragment_movies,
container, false);

//        int rCheck =
ContextCompat.checkSelfPermission(getActivity(),
Manifest.permission.READ_EXTERNAL_STORAGE);
//        if (rCheck != PackageManager.PERMISSION_GRANTED) {
//            ActivityCompat.requestPermissions(getActivity(),
//                    new String[]
{Manifest.permission.READ_EXTERNAL_STORAGE},
REQUEST_READ_EXTERNAL_STORAGE);
//
//        }

        Gson gson = new Gson();

        TextView nothingFound =
root.findViewById(R.id.nothingFound);
        nothingFound.setVisibility(View.VISIBLE);

        nothingFound.setVisibility(View.INVISIBLE);

        RecyclerView list =
root.findViewById(R.id.noMoviesMessage);
        MoviesAdapter adapter_movie = new MoviesAdapter(this,
this.movie_list, this.main_title);
        list.setAdapter(adapter_movie);
        list.setLayoutManager(new
LinearLayoutManager(getActivity()));
```

```
        ProgressBar spinner =
(ProgressBar)root.findViewById(R.id.progressBarMov);
        spinner.setVisibility(ProgressBar.INVISIBLE);

        SearchView searchBar = (SearchView)
root.findViewById(R.id.searchBar);
        searchBar.setOnQueryTextListener(new
SearchView.OnQueryTextListener() {

            @Override
            public boolean onQueryTextSubmit(String query) {

                spinner.setVisibility(ProgressBar.VISIBLE);
                new Thread(new Runnable() {
                @Override
                public void run() {
                    if (query.length() >= 3) {
                        String formattedUrlString =
"http://www.omdbapi.com/?apikey=%s&s=%s&page=1";
                        String apiKey = "7e9fe69e";
                        Queue<String> queue = new LinkedList<>();
                        Requester req = new Requester(queue,
formattedUrlString, apiKey, query);
                        Thread th1 = new Thread(req, "movies");
                        th1.start();
                        try {
                            th1.join();
                            setJSONResponse(queue.remove());
                        } catch (InterruptedException e) {
                            e.printStackTrace();
                        }

                        System.out.println(response);
                        if (response != null &&
                                !gson.fromJson(response,
JsonObject.class).has("Error")) {

nothingFound.setVisibility(View.INVISIBLE);

                            JsonObject gsontmp =
gson.fromJson(response, JsonObject.class);
                            movie_list.clear();

movie_list.addAll(gson.fromJson(gsontmp.get("Search"),
listOfMoviesItemsType));

System.out.println(movie_list.hashCode());
```

```
                            main_title.clear();
                            for (MovieItem movie: movie_list) {
                                main_title.add(movie.getTitle());
                            }
                        } else {
                            movie_list.clear();
                            main_title.clear();
                        }
                    } else {

                        movie_list.clear();
                        main_title.clear();
                    }
                    spinner.setVisibility(ProgressBar.INVISIBLE);

                    getActivity().runOnUiThread(new Runnable() {
                        @Override
                        public void run() {
                            if (query.length() < 3)

nothingFound.setVisibility(View.VISIBLE);
                            adapter_movie.notifyDataSetChanged();
                        }
                    });
                }
            }).start();

//                System.out.println(main_title);
                return false;
            }

            @Override
            public boolean onQueryTextChange(String newText) {
                return true;
            }
        });

//--------This Activity is still present if needed--------
//        Button addMovieButton = (Button)
root.findViewById(R.id.addItem);
//        addMovieButton.setOnClickListener(new
View.OnClickListener() {
//            @Override
//            public void onClick(View v) {
//                Intent intent = new Intent(getActivity(),
AddMovieActivity.class);
//                startActivityForResult(intent, 1);
//                adapter_movie.notifyDataSetChanged();
//              }
//          });

        return root;
    }

    public static void setJSONResponse(String JSON) {
        response = JSON;
    }

    private void updateJSON(String newData) {
        Gson gson = new Gson();
        Type listOfMoviesItemsType = new
TypeToken<ArrayList<MovieItem>>() {}.getType();
        ArrayList<MovieItem> new_movie = gson.fromJson(newData,
listOfMoviesItemsType);
        movie_list.addAll(gson.fromJson(newData,
listOfMoviesItemsType));
        main_title.add(new_movie.get(0).getTitle());
    }

    @Override
    public void onActivityResult(int requestCode, int resultCode,
Intent data) {
        super.onActivityResult(requestCode, resultCode, data);
        if (requestCode == 1) {
            if (resultCode == Activity.RESULT_OK) {
                String returnValue = data.getStringExtra("movie");
                updateJSON(returnValue);
            }
        }
    }


//Not needed anymore
//    public String ReadTextFile(String name) throws IOException {
//        StringBuilder string = new StringBuilder();
//        String line = "";
//        InputStream is = getContext().getAssets().open(name);
//        BufferedReader reader = new BufferedReader(new
InputStreamReader(is));
//        while (true) {
//            try {
//                if ((line = reader.readLine()) == null) break;
//            }
//            catch (IOException e) {
//                e.printStackTrace();
//            }
//            string.append(line);
//        }
//        is.close();
//        return string.toString();
//
//    }
}
```

---Requester.java---

```
package ua.kpi.comsys.iv8106.requester;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.net.HttpURLConnection;
import java.net.MalformedURLException;
import java.net.URL;
import java.util.Queue;

public class Requester implements Runnable {
    private final String formattedUrlString;
    private final String[] parameters;
    private Queue<String> queue;

    public Requester(Queue<String> queue, String
formattedUrlString, String... parameters) {
        this.queue = queue;
        this.formattedUrlString = formattedUrlString;
        this.parameters = parameters;
    }

    @Override
    public void run() {
        String res = sendRequest(formattedUrlString, parameters);
        queue.add(res);
    }

    private String sendRequest(String formattedUrlString, String[]
parameters) {
        try {
            URL url = new
URL(String.format(formattedUrlString,parameters));

System.out.println(String.format(formattedUrlString,parameters));
            HttpURLConnection connection = (HttpURLConnection)
url.openConnection();
```

```
            connection.setRequestProperty("accept",
"application/json");
            InputStream responseStream =
connection.getInputStream();

            InputStreamReader isReader = new
InputStreamReader(responseStream);
            BufferedReader reader = new BufferedReader(isReader);
            StringBuilder textBuilder = new StringBuilder();
            String line;
            while((line = reader.readLine())!= null){
                textBuilder.append(line);
            }
            connection.disconnect();
            return textBuilder.toString();

        } catch (MalformedURLException e) {
            e.printStackTrace();

        } catch (IOException e) {
            e.printStackTrace();
        }
        return null;
    }
}
```

---ImageItem.java---

```
package ua.kpi.comsys.iv8106.model;

import android.graphics.Bitmap;
import android.net.Uri;

public class ImageItem {
    private boolean isVisible = true;

    private Bitmap bitmap = null;

    public void setBitmap(Bitmap bitmap) {
        this.bitmap = bitmap;
    }

    public Bitmap getBitmap() {
        return bitmap;
    }

    private Uri     imgpath;

    private String id;
    private String pageURL;
    private String type;
    private String tags;
    private String previewURL;
    private String previewWidth;
    private String previewHeight;
    private String webformatURL;
    private String webformatWidth;
    private String webformatHeight;
    private String largeImageURL;
    private String imageWidth;
    private String imageHeight;
    private String imageSize;
    private String views;
    private String downloads;
    private String favorites;
    private String likes;
    private String comments;
    private String user_id;
    private String user;
    private String userImageURL;

    public boolean isVisible() {
        return isVisible;
    }

    public Uri getImgpath() {
        return imgpath;
    }

    public String getId() {
        return id;
    }

    public String getPageURL() {
        return pageURL;
    }

    public String getType() {
        return type;
    }

    public String getTags() {
        return tags;
    }

    public String getPreviewURL() {
        return previewURL;
    }

    public String getPreviewWidth() {
        return previewWidth;
    }

    public String getPreviewHeight() {
        return previewHeight;
    }

    public String getWebformatURL() {
        return webformatURL;
    }

    public String getWebformatWidth() {
        return webformatWidth;
    }

    public String getWebformatHeight() {
        return webformatHeight;
    }

    public String getLargeImageURL() {
        return largeImageURL;
    }

    public String getImageWidth() {
        return imageWidth;
    }

    public String getImageHeight() {
        return imageHeight;
    }

    public String getImageSize() {
        return imageSize;
    }

    public String getViews() {
        return views;
    }

    public String getDownloads() {
        return downloads;
    }

    public String getFavorites() {
        return favorites;
    }

    public String getLikes() {
        return likes;
    }

    public String getComments() {
        return comments;
    }

    public String getUser_id() {
        return user_id;
    }

    public String getUser() {
        return user;
    }

    public String getUserImageURL() {
        return userImageURL;
    }

}
```

4. Висновок:

Було створено програму за завданням, навчився робити запити та обробляти відповіді з сервера, такж було повторено колбеки та потоки, їх було застосовано при завантаженні даних, щоб не зупинявся головний процес та для виконання дій після завершення обробки.