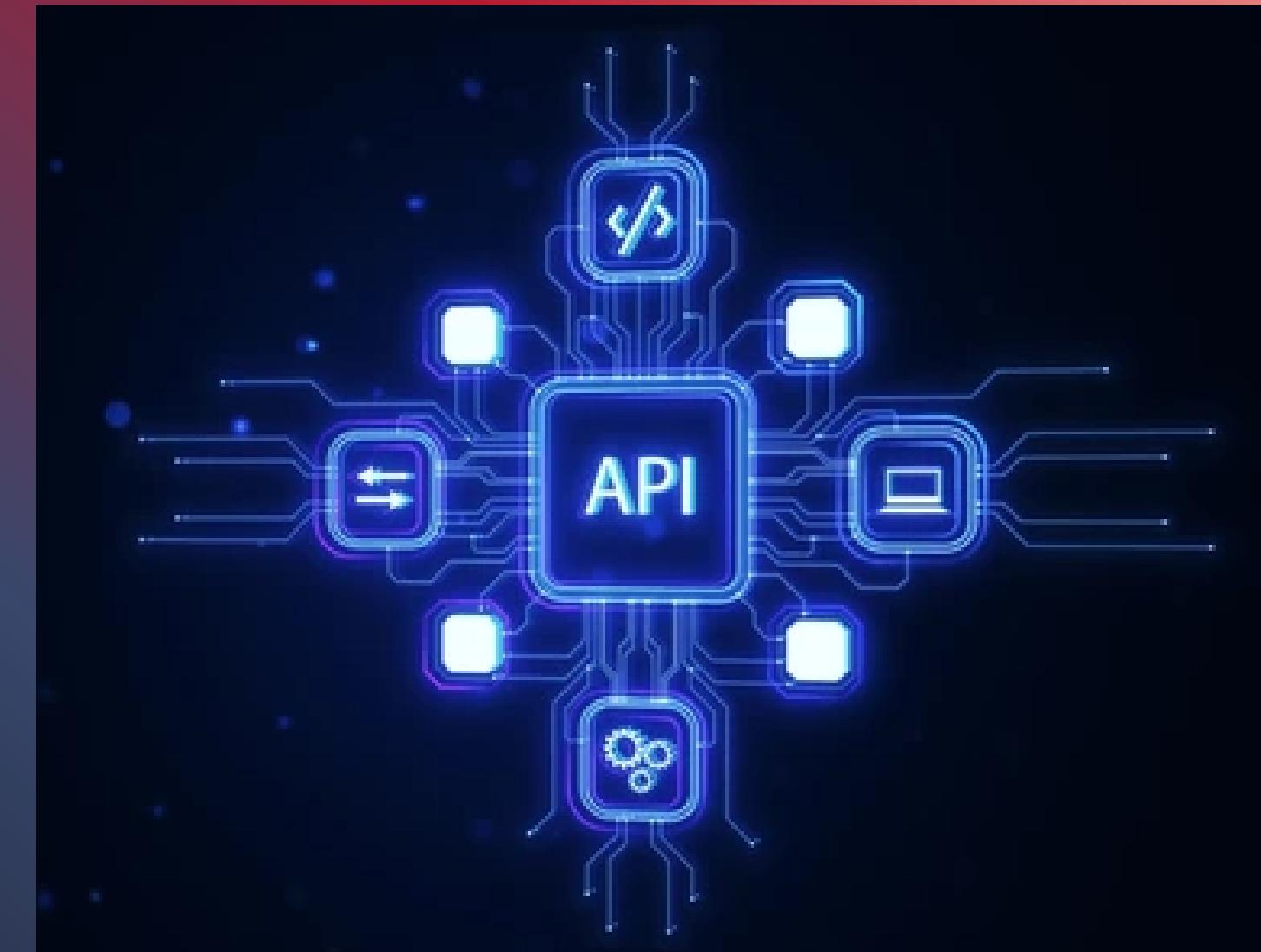


APIs remotas (REST/GraphQL)

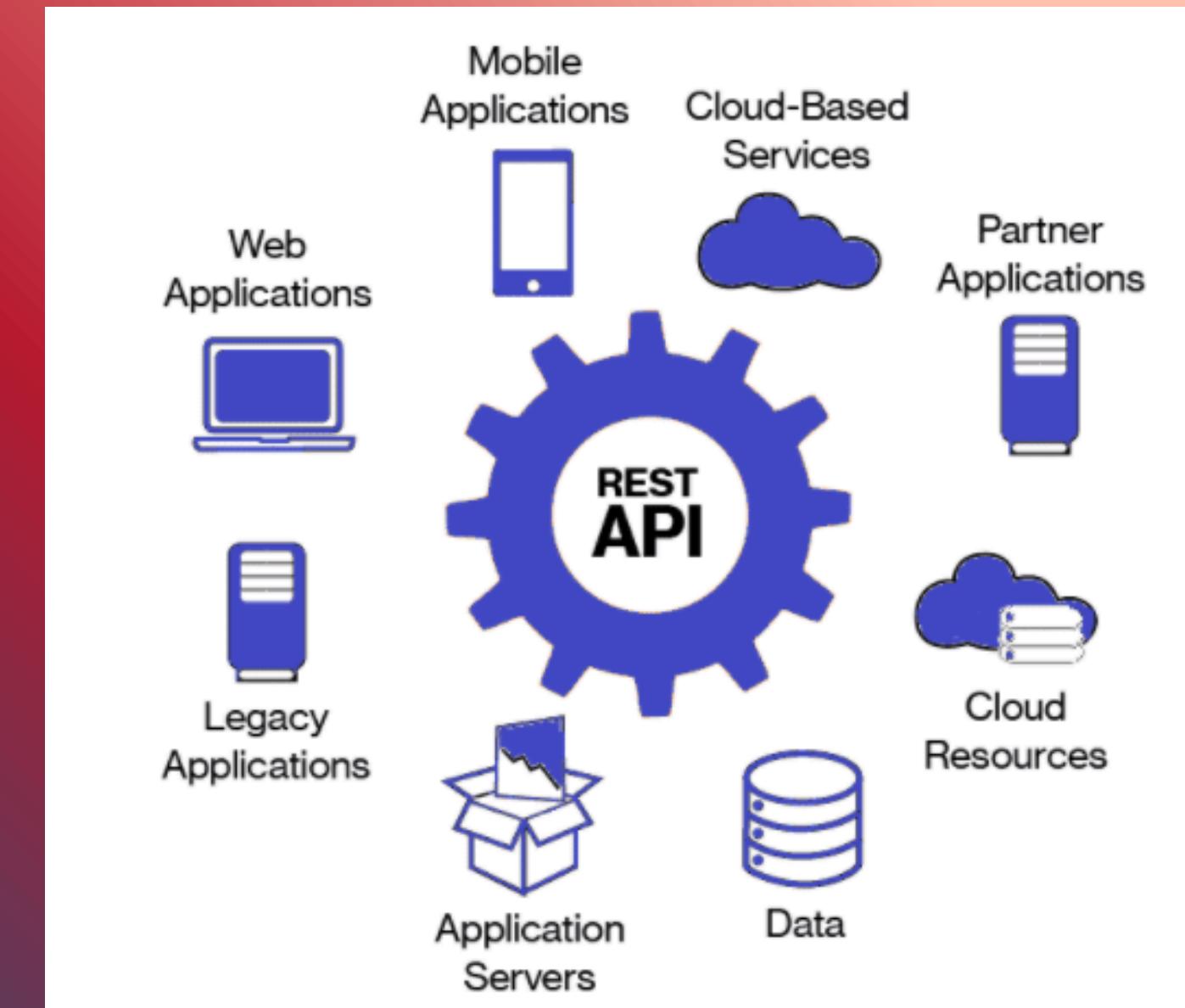


Elementos:

- Álvaro Frias - 49400;
- Eduardo Mesquita - 49507;
- Ari Jesus - 49903;
- Tiago Valério - 52334.

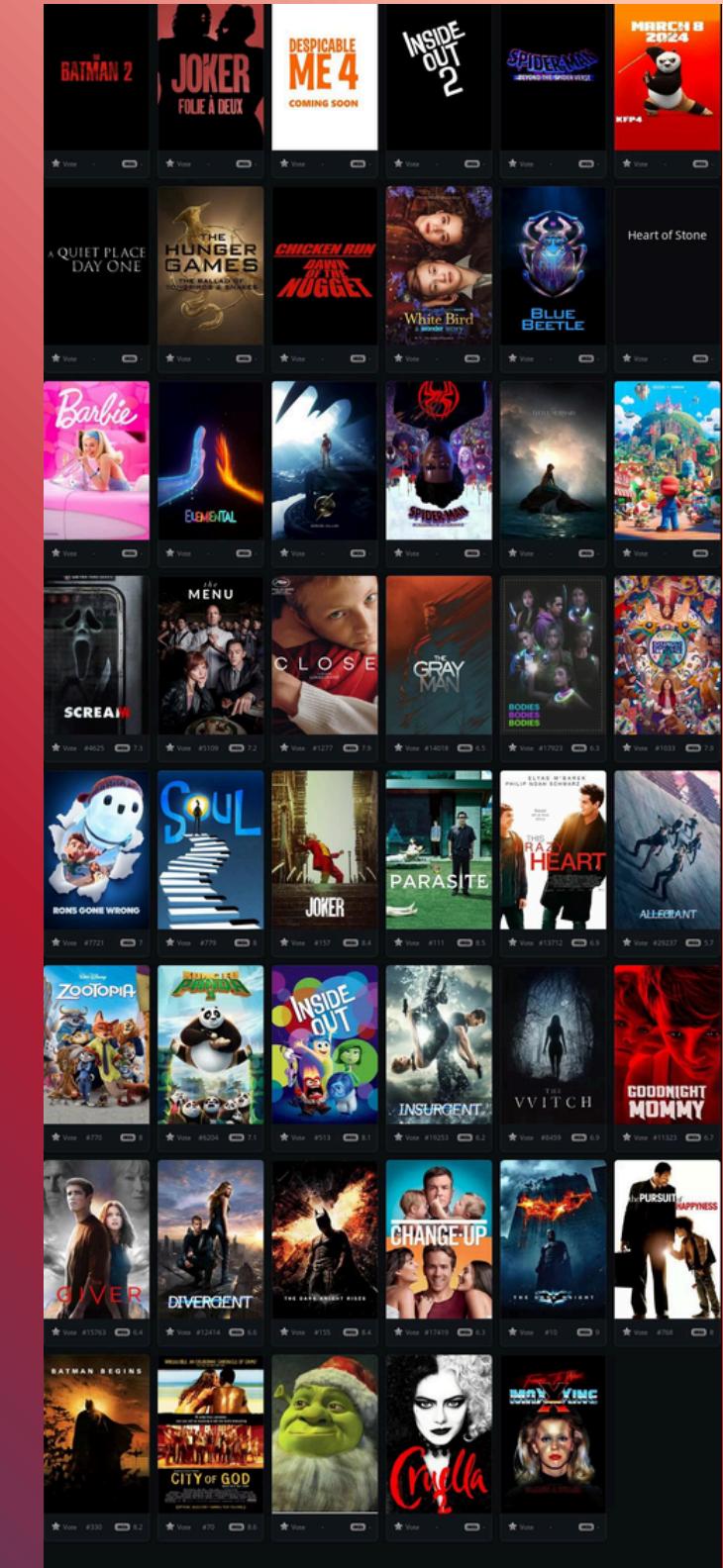
Introdução:

API REST é um conjunto de regras que permite a comunicação entre sistemas via HTTP, sem manter estado entre solicitações.



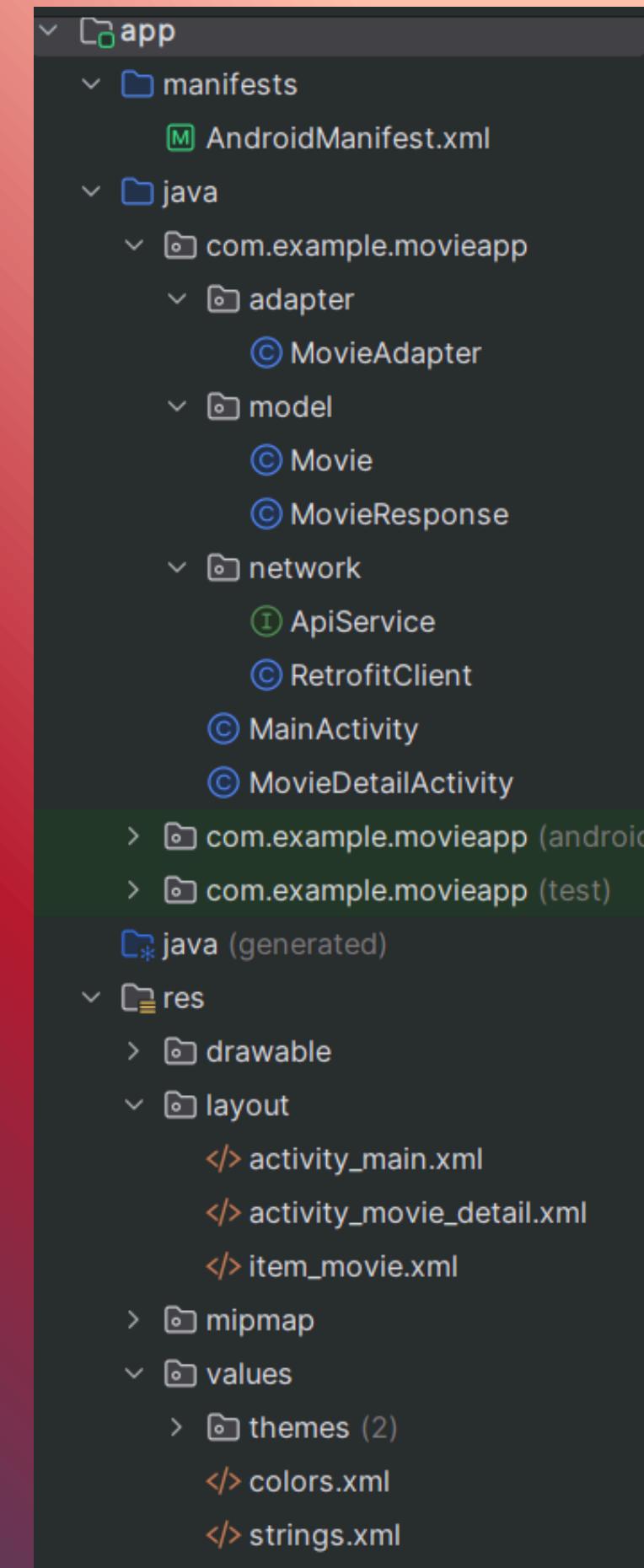
Objetivo do Trabalho:

- Criar uma aplicação Android em Java que apresente uma lista de filmes;
- Consumir dados de uma API remota (TMDB);
- Apresentar os dados em tempo real;
- Demonstrar integração RESTful.



Estrutura da Aplicação

- model/ → Estrutura dos dados (JSON → objetos Java)
- network/ → Comunicação REST (Retrofit + API Key)
- adapter/ → Ligação entre dados e interface
- ui/ e res/layout/ → Interface visual e interação



Código da Integração REST (Imagens de exemplo)

```
package com.example.movieapp.network;

import com.example.movieapp.model.MovieResponse;
import retrofit2.Call;
import retrofit2.http.GET;
import retrofit2.http.Query;

3 usages
public interface ApiService {
    1 usage
    @GET("movie/popular")
    Call<MovieResponse> getPopularMovies(@Query("api_key") String apiKey, @Query("page") int page);
}
```

Interface ApiService define os endpoints

```
package com.example.movieapp.network;

import okhttp3.HttpUrl;
import okhttp3.Interceptor;
import okhttp3.OkHttpClient;
import okhttp3.Request;
import okhttp3.Response;
import retrofit2.Retrofit;
import retrofit2.converter.gson.GsonConverterFactory;

import java.io.IOException;

2 usages
public class RetrofitClient {
    1 usage
    private static final String BASE_URL = "https://api.themoviedb.org/3/";
    1 usage
    private static final String API_KEY = "938f7d1bef2541960231fabd155a12ee";
    3 usages
    private static Retrofit retrofit;

    public static Retrofit getClient() {
        if (retrofit == null) {
            OkHttpClient client = new OkHttpClient.Builder()
                .addInterceptor(new Interceptor() {
                    @Override
                    public Response intercept(Chain chain) throws IOException {
                        Request original = chain.request();
                        HttpUrl originalHttpUrl = original.url();

                        HttpUrl url = originalHttpUrl.newBuilder()
                            .addQueryParameter("name", "api_key", API_KEY)
                            .build();

                        Request.Builder requestBuilder = original.newBuilder().url(url);
                        Request request = requestBuilder.build();
                        return chain.proceed(request);
                    }
                })
                .build();

            retrofit = new Retrofit.Builder()
                .baseUrl(BASE_URL)
                .addConverterFactory(GsonConverterFactory.create())
                .client(client)
                .build();
        }
        return retrofit;
    }
}
```

Retrofit configura base URL e API key

Código da Interface (Imagens de Exemplo)

```
package com.example.movieapp;

import ...

public class MainActivity extends AppCompatActivity {
    3 usages
    private RecyclerView rvMovies;
    2 usages
    private MovieAdapter adapter;
    5 usages
    private String apiKey;
    2 usages
    private String imageUrl;
    1 usage
    private String imageBaseUrl;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        rvMovies = findViewById(R.id.rvMovies);
        rvMovies.setLayoutManager(new LinearLayoutManager(this));

        apiKey = getString(R.string.tmdb_api_key);
        imageUrl = getString(R.string.image_base_url);

        if (apiKey == null || apiKey.isEmpty() || apiKey.contains("YOUR_TMDB_API_KEY")) {
            Toast.makeText(context, "Insira a sua TMDB API key em res/values/strings.xml", Toast.LENGTH_LONG).show();
            return;
        }

        fetchPopularMovies();
    }
}
```

MainActivity recebe os dados da API

```
1 usage
private void fetchPopularMovies() {
    ApiService apiService = RetrofitClient.getClient().create(ApiService.class);
    Call<MovieResponse> call = apiService.getPopularMovies(apiKey, page: 1);
    call.enqueue(new Callback<MovieResponse>() {
        @Override
        public void onResponse(Call<MovieResponse> call, Response<MovieResponse> response) {
            if (response.isSuccessful() && response.body() != null) {
                List<Movie> movies = response.body().getResults();
                adapter = new MovieAdapter(context: MainActivity.this, movies, imageUrl);
                rvMovies.setAdapter(adapter);
            } else {
                Toast.makeText(context: MainActivity.this, text: "Erro ao obter filmes", Toast.LENGTH_LONG).show();
                Log.e(tag: "MainActivity", msg: "Response error: " + response.message());
            }
        }

        @Override
        public void onFailure(Call<MovieResponse> call, Throwable t) {
            Toast.makeText(context: MainActivity.this, text: "Falha: " + t.getMessage(), Toast.LENGTH_LONG).show();
            Log.e(tag: "MainActivity", msg: "Failure: ", t);
        }
    });
}
```

```
package com.example.movieapp.adapter;

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

import androidx.annotation.NonNull;
import androidx.recyclerview.widget.RecyclerView;

import com.bumptech.glide.Glide;
import com.example.movieapp.MovieDetailActivity;
import com.example.movieapp.R;
import com.example.movieapp.model.Movie;

import java.util.List;

4 usages
public class MovieAdapter extends RecyclerView.Adapter<MovieAdapter.MovieViewHolder> {
    5 usages
    private final Context ctx;
    4 usages
    private final List<Movie> movies;
    1 usage
    private final String imageUrl;

    1 usage
    public MovieAdapter(Context ctx, List<Movie> movies, String imageUrl) {
        this.ctx = ctx;
        this.movies = movies;
        this.imageUrl = imageUrl;
    }

    @NonNull
    @Override
    public MovieViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
        View view = LayoutInflater.from(ctx).inflate(R.layout.item_movie, parent, attachToRoot: false);
        return new MovieViewHolder(view);
    }

    @Override
    public void onBindViewHolder(@NonNull MovieViewHolder holder, int position) {
        Movie m = movies.get(position);

        // Defensive checks
        String title = m.getTitle() != null ? m.getTitle() : "Sem título";
        String overview = m.getOverview() != null ? m.getOverview() : "";
        String posterPath = m.getPosterPath();

        holder.textViewTitle.setText(title);
        holder.textViewOverview.setText(overview);

        String posterUrl = "https://image.tmdb.org/t/p/w500" + m.getPosterPath();
        Glide.with(ctx).RequestManager
            .load(posterUrl).RequestBuilder<Drawable>
            .centerCrop()
            .placeholder(android.R.drawable.ic_menu_report_image)
            .into(holder.imageViewPoster);

        holder.itemView.setOnClickListener(v -> {
            Intent i = new Intent(ctx, MovieDetailActivity.class);
            i.putExtra(name: "title", m.getTitle());
            i.putExtra(name: "overview", m.getOverview());
            i.putExtra(name: "poster_path", m.getPosterPath());
            i.putExtra(name: "rating", m.getVoteAverage());
            ctx.startActivity(i);
        });
    }

    @Override
    public int getItemCount() {
        return movies != null ? movies.size() : 0;
    }

    4 usages
    static class MovieViewHolder extends RecyclerView.ViewHolder {
        2 usages
        ImageView imageViewPoster;
        2 usages
        TextView textViewTitle;
        2 usages
        TextView textViewOverview;

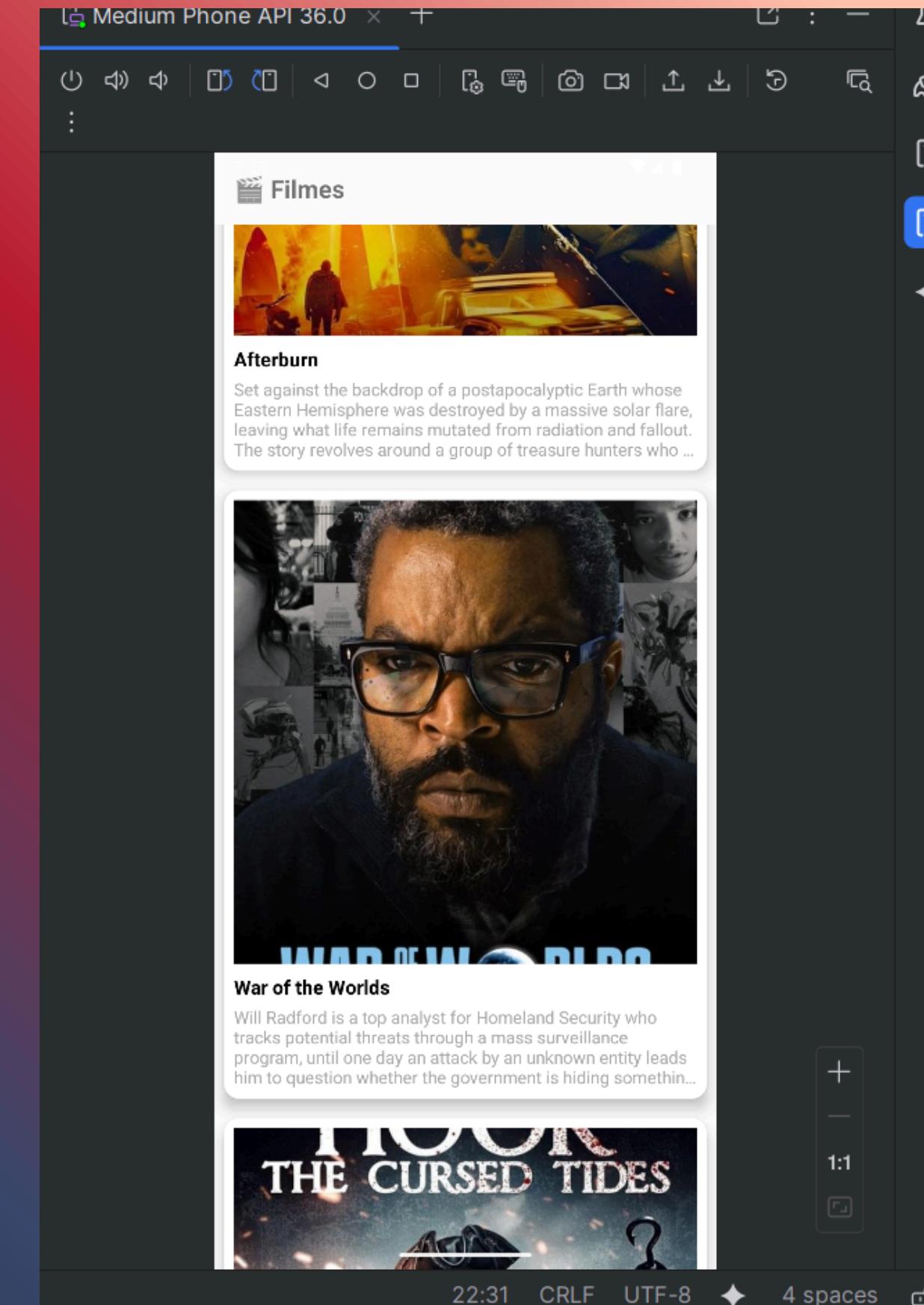
        1 usage
        MovieViewHolder(@NonNull View itemView) {
            super(itemView);
            imageViewPoster = itemView.findViewById(R.id.imageViewPoster);
            textViewTitle = itemView.findViewById(R.id.textViewTitle);
            textViewOverview = itemView.findViewById(R.id.textViewOverview);
        }
    }
}
```

MovieAdapter mostra cada filme no RecyclerView

Interface da Aplicação (tipos de dados exemplo)

Resultados Obtidos:

- Lista dinâmica de filmes;
- Carregamento de imagens e descrições;
- Detalhes de cada filme ao clicar.



Conclusão

Conclusões e Aprendizagens:

- Integração com API REST implementada com sucesso;
- Comunicação com servidor TMDB via Retrofit;
- Manipulação de JSON e exibição dinâmica;
- Aprendizagem: arquitetura em camadas, consumo de dados online;
- Possível melhoria: sistema de favoritos ou pesquisa.



The diagram shows a central smartphone connected to four colored clouds, each representing a different API category. The red cloud on the left contains icons for social media (Twitter bird), messaging (@ symbol), email (envelope), music (note), and video (film). The orange cloud in the middle contains icons for file sharing (cloud with arrows), people (team), images (camera), and location (map). The green cloud on the right contains icons for file storage (folder), security (padlock), and system (gear, wrench, Wi-Fi). The yellow cloud at the bottom contains icons for system (gear, wrench) and connectivity (Wi-Fi).

SMTP & API

Generate a new API key

SMTP API Keys

Version	API Key	Name	Created on
v3	*****Xo4rJT	prestashop 1.7	April 4, 2023 12:48 PM
v3	*****QnbN3k	WordPress	

Objetivos:



Objetivos e Implementações Futuras:

- API do Mapbox para representação dos locais
- MediaWiki API para recolha de informações para o uso no nosso trabalho coletivo
- Entre outros...