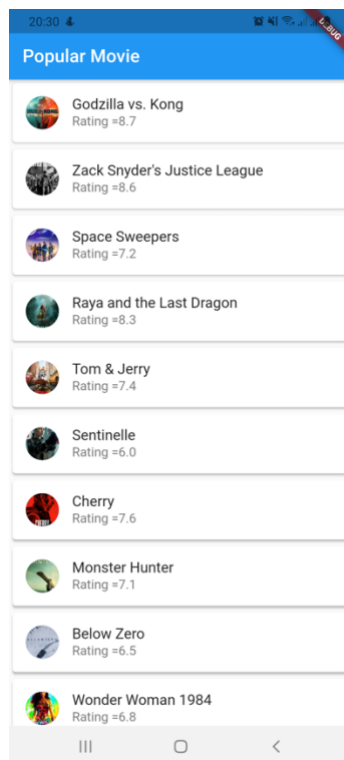


BAB 7. HTTP REQUEST

7.1 Desain Aplikasi

Pada project kali ini anda akan membuat sebuah aplikasi yang mengakses REST API dari themoviedb.org. Aplikasi yang dibuat akan menampilkan film populer dan menampilkan detail dari film populer yang di pilih oleh user. Hasil akhir dari aplikasi yang dibuat dapat dilihat pada gambar berikut ini.



Desain yang dibuat mengikuti pola master detail pada praktikum 5 tetapi sekarang berisi data dari REST API.

7.2 Persiapan Project

Untuk membuat project ini anda membutuhkan :

1. Api Key dari themoviedb.org untuk mendapatkan api key dapat mengikuti tutorial berikut ini :
 - a. <https://www.dicoding.com/blog/registrasi-testing-themoviedb-api/>
 - b. <https://stackoverflow.com/questions/31047815/api-key-for-themoviedb-org>
2. Library Flutter http
 - a. Setelah project flutter berhasil dibuat tambahkan library http ke project tersebut. Untuk menambahkan library bisa menggunakan plugin vscode pubspec assist atau mengikuti tutorial disini

- b. <https://stackoverflow.com/questions/59915827/how-to-add-a-library-to-flutter-dependencies>

3. Jika ada permintaan update flutter lakukan flutter upgrade.

7.3 Koneksi ke REST API

7.3.1 Menguji koneksi di dashboard api themoviedb

Untuk menguji koneksi ke REST API themoviedb dapat dilakukan di link berikut ini

<https://developers.themoviedb.org/3/movies/get-popular-movies> pada link berikut anda dapat menguji dan membuat request ke server themoviedb dan melihat response nya dalam berbagai format json.

The screenshot shows the TMDB API dashboard for the 'Get Popular' endpoint. On the left is a sidebar menu with categories like GETTING STARTED, ACCOUNT, AUTHENTICATION, etc. The main area is titled 'Get Popular' and shows the endpoint 'GET /movie/popular'. Below this, there's a 'Variables' section with a table for 'api_key' (Your TMDb API key, optional). The 'Query String' section has a table with variables: 'api_key' (required), 'language' (en-US, optional), 'page' (1, optional), and 'region' (String, optional). At the bottom, there's a 'SEND REQUEST' button and the resulting URL: 'https://api.themoviedb.org/3/movie/popular?api_key=<<api_key>>&language=en-US&page=1'.

Variable	Value	Required
api_key	Your TMDb API key	optional

Variable	Value	Required
api_key	<<api_key>>	required
language	en-US	optional
page	1	optional
region	String	optional

SEND REQUEST `https://api.themoviedb.org/3/movie/popular?api_key=<<api_key>>&language=en-US&page=1`

7.3.2 Mengkoneksikan Aplikasi Flutter ke themoviedb dengan package http

Untuk mulai mengkoneksikan aplikasi flutter yang dibuat ke rest api themoviedb.org lakukanlah langkah berikut:

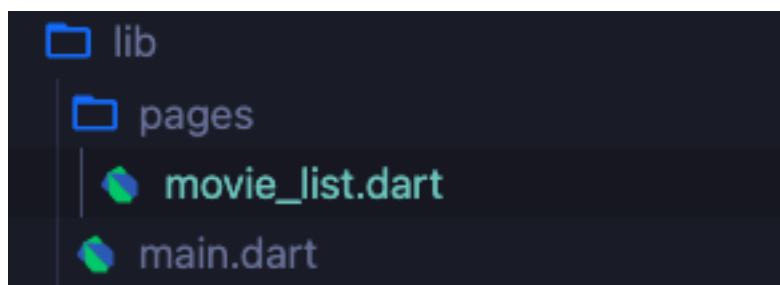
1. Untuk mengakses internet kita harus menambahkan permission internet pada android manifest cari lah file android manifest.xml pada folder android/app/src/main/AndroidManifest.xml kemudian tambahkan permission internet.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android" package="com.example.moviedb2">
    <uses-permission android:name="android.permission.INTERNET" />
    <application android:label="moviedb2" android:icon="@mipmap/ic_launcher">
        <activity android:name=".MainActivity" android:launchMode="singleTop" android:theme="@style/LaunchTheme" android:configChanges="orientation|screenSize|uiMode"
            android:exported="true">
            <!-- Specifies an Android theme to apply to this Activity as soon as
                 the Android process has started. This theme is visible to the user
                 while the Flutter UI initializes. After that, this theme continues
                 to determine the Window background behind the Flutter UI. -->
            <meta-data android:name="io.flutter.embedding.android.NormalTheme" android:resource="@style/NormalTheme" />
            <!-- Displays an Android View that continues showing the launch screen
                 Drawable until Flutter paints its first frame, then this splash
            <!-- Displays an Android View that continues showing the launch screen
                 Drawable until Flutter paints its first frame, then this splash
        </activity>
    </application>
</manifest>
```

2. Hapus komentar yang ada pada file main.dart dari kode program awal.
3. Ubah MyHomePage menjadi stateless widget.

```
1  import 'package:flutter/material.dart';
2
3  Run | Debug
4  void main() {
5    runApp(MyApp());
6  }
7
8  class MyApp extends StatelessWidget {
9    // This widget is the root of your application.
10   @override
11   Widget build(BuildContext context) {
12     return MaterialApp(
13       title: 'Flutter Demo',
14       theme: ThemeData(
15         primarySwatch: Colors.blue,
16       ), // ThemeData
17       home: MyHomePage(),
18     ); // MaterialApp
19   }
20 }
21
22 class MyHomePage extends StatelessWidget {
23   @override
24   Widget build(BuildContext context) {
25     return Container();
26   }
27 }
```

4. Buat file baru baru “pages” pada folder lib dan buat sebuah file dengan nama movie_list.dart.



5. Buat sebuah statefull widget pada file movie_list.dart

```

1  import 'package:flutter/material.dart';
2
3  class MovieList extends StatefulWidget {
4      @override
5      _MovieListState createState() => _MovieListState();
6  }
7
8  class _MovieListState extends State<MovieList> {
9      @override
10     Widget build(BuildContext context) {
11         return Container();
12     }
13 }
14

```

6. Import File movie_list.dart ke main.dart dan gunakan movie_list.dart sebagai return dari class MyHomePage

```

22 class MyHomePage extends StatelessWidget {
23     @override
24     Widget build(BuildContext context) {
25         return MovieList();
26     }
27 }

```

7. Sebelum melanjutkan ke pembuatan aplikasi kita buat terlebih dahulu sebuah helper class untuk konek ke rest api themoviedb. Buat lah sebuah folder dengan nama service dan isi dengan file http_service.dart.

```

4  import 'package:http/http.dart' as http;
5
6  class HttpService {
7      final String apiKey = 'isi dengan api key anda';
8      final String baseUrl = 'https://api.themoviedb.org/3/movie/popular?api_key=';
9  }

```

8. Selanjutnya buat sebuah function untuk mengambil response dari server themoviedb.org

```

1  import 'dart:io';
2  import 'package:http/http.dart' as http;
3
4  class HttpService {
5      final String apiKey = 'isi dengan api key anda';
6      final String baseUrl = 'https://api.themoviedb.org/3/movie/popular?api_key=';
7
8      Future<String> getPopularMovies() async {
9          final String uri = baseUrl + apiKey;
10
11          http.Response result = await http.get(Uri.parse(uri));
12          if (result.statusCode == HttpStatus.ok) {
13              print("Sukses");
14              String response = result.body;
15              return response;
16          } else {
17              print("Fail");
18              return null;
19          }
20      }
21  }

```

9. Update file movie_list.dart agar dapat menggunakan file httpService dan mereturn kan widget scaffold.
10. Pada function _MovieListState tambahkan variabel berikut ini untuk variabel service jangan lupa import dulu file httpservice nya.

```

9  class _MovieListState extends State<MovieList> {
10      String result = "";
11      HttpService service;
12

```

11. Kemudian tambahkan method override init state agar permintaan ke rest api dapat dilakukan ketika state di inisialisasi.

```

9  class _MovieListState extends State<MovieList> {
10      String result = "";
11      HttpService service;
12
13      @override
14      void initState() {
15          service = HttpService();
16          super.initState();
17      }
18

```

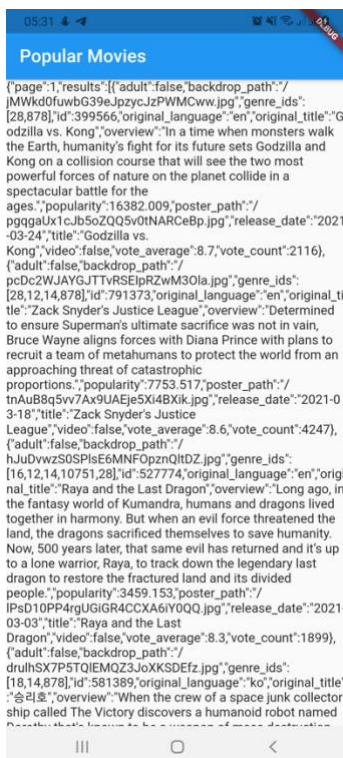
12. Selanjutnya update function build pada movieListState dengan menggunakan widget scaffold.

```

19  @override
20  Widget build(BuildContext context) {
21    service.getPopularMovies().then((value) => {
22      setState(() {
23        result = value;
24      })
25    });
26    return Scaffold(
27      appBar: AppBar(
28        title: Text("Popular Movies"),
29      ), // AppBar
30      body: Container(
31        child: Text(result),
32      ), // Container
33    ); // Scaffold
34  }
35  }

```

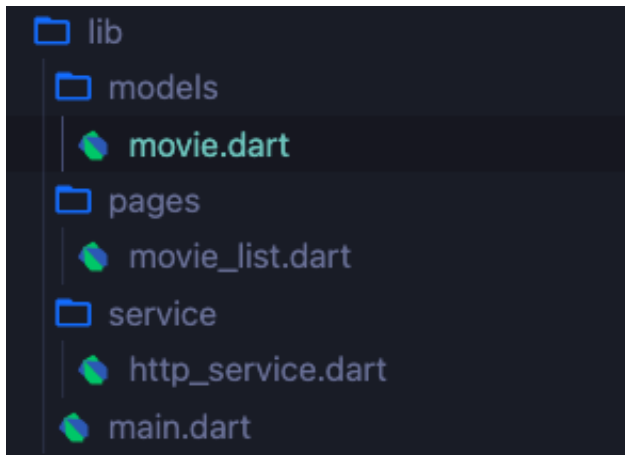
13. Jalankan aplikasi, jika di awal terjadi error reload menggunakan hot restart.



14. Pada langkah ini kita sudah berhasil mendapatkan response dari REST Api ke widget text namun belum tampil dengan baik karena belum menggunakan listview dan models. Selanjutnya kita persiapkan sebuah file models dan listview.

7.3.3 Membuat model untuk response http

1. Buat folder models didalam folder lib dan isikan dengan file movie.dart



2. Di dalam movie.dart buat lah sebuah class movie lengkap dengan variabel dan konstruktor seperti dibawah ini.

```
lib > models > movie.dart > ...  
1 class Movie {  
2   int id;  
3   String title;  
4   double voteAverage;  
5   String overview;  
6   String posterPath;  
7  
8   Movie(this.id, this.title, this.voteAverage, this.overview, this.posterPath);  
9 }
```

3. Selanjutnya buatlah sebuah function untuk mengkonversi json menjadi response yang sesuai dengan class movie.

```
lib > models > movie.dart > ...  
1 class Movie {  
2   int id;  
3   String title;  
4   double voteAverage;  
5   String overview;  
6   String posterPath;  
7  
8   Movie(this.id, this.title, this.voteAverage, this.overview, this.posterPath);  
9  
10  Movie.fromJson(Map<String, dynamic> parsedJson) {  
11    this.id = parsedJson['id'];  
12    this.title = parsedJson['title'];  
13    this.voteAverage = parsedJson['vote_average'] * 1.0;  
14    this.overview = parsedJson['overview'];  
15    this.posterPath = parsedJson['poster_path'];  
16  }  
17 }
```

4. Selanjutnya update function http_service.dart menjadi seperti dibawah ini.

```

1  import 'dart:convert';
2  import 'dart:io';
3  import 'package:http/http.dart' as http;
4  import 'package:moviedb2/models/movie.dart';
5
6  class HttpService {
7    final String apiKey = 'api key';
8    final String baseUrl = 'https://api.themoviedb.org/3/movie/popular?api_key=';
9
10   Future<List> getPopularMovies() async {
11     final String uri = baseUrl + apiKey;
12
13     http.Response result = await http.get(Uri.parse(uri));
14     if (result.statusCode == HttpStatus.ok) {
15       print("Sukses");
16       final jsonResponse = json.decode(result.body);
17       final moviesMap = jsonResponse['results'];
18       List movies = moviesMap.map((i) => Movie.fromJson(i)).toList();
19       return movies;
20     } else {
21       print("Fail");
22       return null;
23     }
24   }
25 }

```

5. Sampai pada tahap ini aplikasi anda tidak dapat berjalan dengan baik, biarkan dan lanjutkan ke langkah selanjutnya untuk membuat list populer movie.

7.4 Membuat halaman list Populer Movie

Untuk membuat list view pada widget movie list anda memerlukan beberapa widget baru antara lain list view dan card. Selain itu juga dibutuhkan data dari http service. Langkah langkah yang perlu dilakukan adalah sebagai berikut :

1. Update inisialisasi variabel pada class movieliststate

```

9  class _MovieListState extends State<MovieList> {
10    int moviesCount;
11    List movies;
12    HttpService service;

```

2. Tambahkan method initialize() pada class movieliststate


```

9  class _MovieListState extends State<MovieList> {
10      int moviesCount;
11      List movies;
12      HttpService service;
13
14      Future initialize() async {
15          movies = [];
16          movies = await service.getPopularMovies();
17          setState(() {
18              moviesCount = movies.length;
19              movies = movies;
20          });
21      }
22

```

3. Tambahkan function initialize pada initState

```

9  class _MovieListState extends State<MovieList> {
10      int moviesCount;
11      List movies;
12      HttpService service;
13
14      Future initialize() async {
15          movies = [];
16          movies = await service.getPopularMovies();
17          setState(() {
18              moviesCount = movies.length;
19              movies = movies;
20          });
21      }
22
23      @override
24      void initState() {
25          service = HttpService();
26          initialize();
27          super.initState();
28      }

```

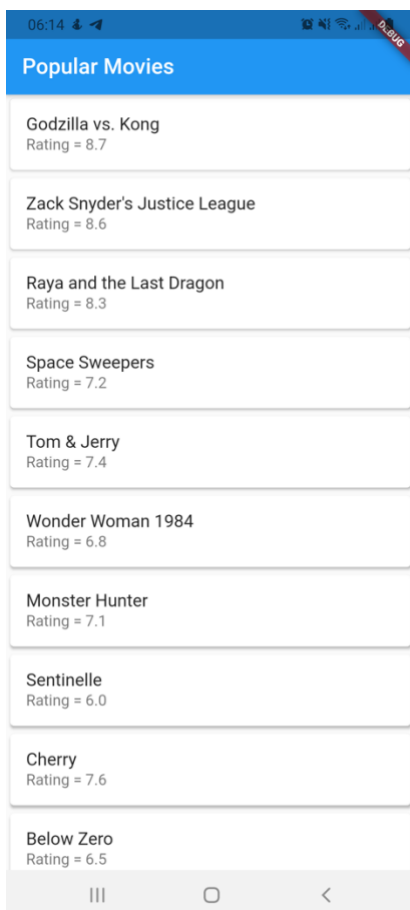
4. Selanjutnya buat listview dan card berdasarkan data dari initialize

```

30 @override
31 Widget build(BuildContext context) {
32   return Scaffold(
33     appBar: AppBar(
34       title: Text("Popular Movies"),
35     ), // AppBar
36     body: ListView.builder(
37       itemCount: (this.moviesCount == null) ? 0 : this.moviesCount,
38       itemBuilder: (context, int position) {
39         return Card(
40           color: Colors.white,
41           elevation: 2.0,
42           child: ListTile(
43             title: Text(movies[position].title),
44             subtitle: Text(
45               'Rating = ' + movies[position].voteAverage.toString(),
46             ), // Text
47           ), // ListTile
48         ); // Card
49       },
50     ); // ListView.builder // Scaffold
51   }

```

5. Lakukan hot restart dan aplikasi akan berubah menjadi seperti dibawah ini.



6. Selamat anda berhasil membuat listview sekarang saatnya mempercantik listview yang dibuat.

7. Challenge : carilah cara menambahkan gambar dari response api ke listview, tambahkan gambar tersebut ke listview.

7.5 Membuat halaman detail Populer Movie

1. Untuk membuat perpindahan dari movie list ke movie detail buatlah onTap event di listview pada movie list.

```
39     return Card(  
40       color: Colors.white,  
41       elevation: 2.0,  
42       child: ListTile(  
43         title: Text(movies[position].title),  
44         subtitle: Text(  
45           'Rating = ' + movies[position].voteAverage.toString(),  
46         ), // Text  
47         onTap: () {  
48           MaterialPageRoute route = MaterialPageRoute(  
49             builder: (_) => MovieDetail(movies[position])); // MaterialPageRoute  
50           Navigator.push(context, route);  
51         },  
52       ), // ListTile  
53     ); // Card
```

2. Pada event on tap Widget MovieDetail belum dibuat, buatlah widget ini pada folder pages/movie_detail.dart.

```
lib > pages > movie_detail.dart > ...  
1  import 'package:flutter/material.dart';  
2  
3  class MovieDetail extends StatelessWidget {  
4    @override  
5    Widget build(BuildContext context) {  
6      return Container();  
7    }  
8  }
```

3. Lengkapi Movie Detail untuk menerima parameter Movies

```

lib > pages > movie_detail.dart > ...
1  import 'package:flutter/material.dart';
2  import 'package:moviedb2/models/movie.dart';
3
4  class MovieDetail extends StatelessWidget {
5      final Movie movie;
6      final String imgPath = 'https://image.tmdb.org/t/p/w500/';
7
8      MovieDetail(this.movie);
9
10     @override
11     Widget build(BuildContext context) {
12         return Container();
13     }
14 }
15

```

4. Lengkapi detail widget.

```

10     @override
11     Widget build(BuildContext context) {
12         String path;
13         if (movie.posterPath != null) {
14             path = imgPath + movie.posterPath;
15         } else {
16             path =
17                 'https://images.freeimages.com/images/large-previews/5eb/movie-clapboard-1184339.jpg';
18         }
19         double height = MediaQuery.of(context).size.height;
20         return Scaffold(
21             appBar: AppBar(
22                 title: Text(movie.title),
23             ), // AppBar
24             body: SingleChildScrollView(
25                 child: Center(
26                     child: Column(
27                         children: [
28                             Container(
29                                 padding: EdgeInsets.all(16),
30                                 height: height / 1.5,
31                                 child: Image.network(path)), // Container
32                             Container(
33                                 child: Text(movie.overview),
34                                 padding: EdgeInsets.only(left: 16, right: 16),
35                             ), // Container
36                         ],
37                     ), // Column
38                 ), // Center
39             ), // SingleChildScrollView
40         ); // Scaffold
41     }

```

5. Challenge Modifikasilah Tampilan agar terlihat lebih menarik