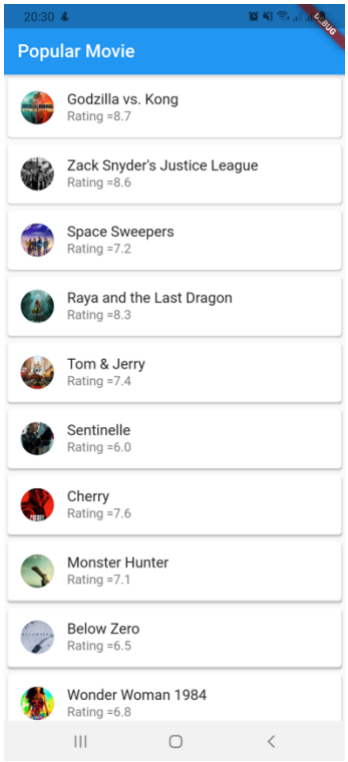
BAB 7. HTTP REQUEST

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 filem populer dan menampilkan detail dari filem 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 assit atau mengikuti tutorial disini

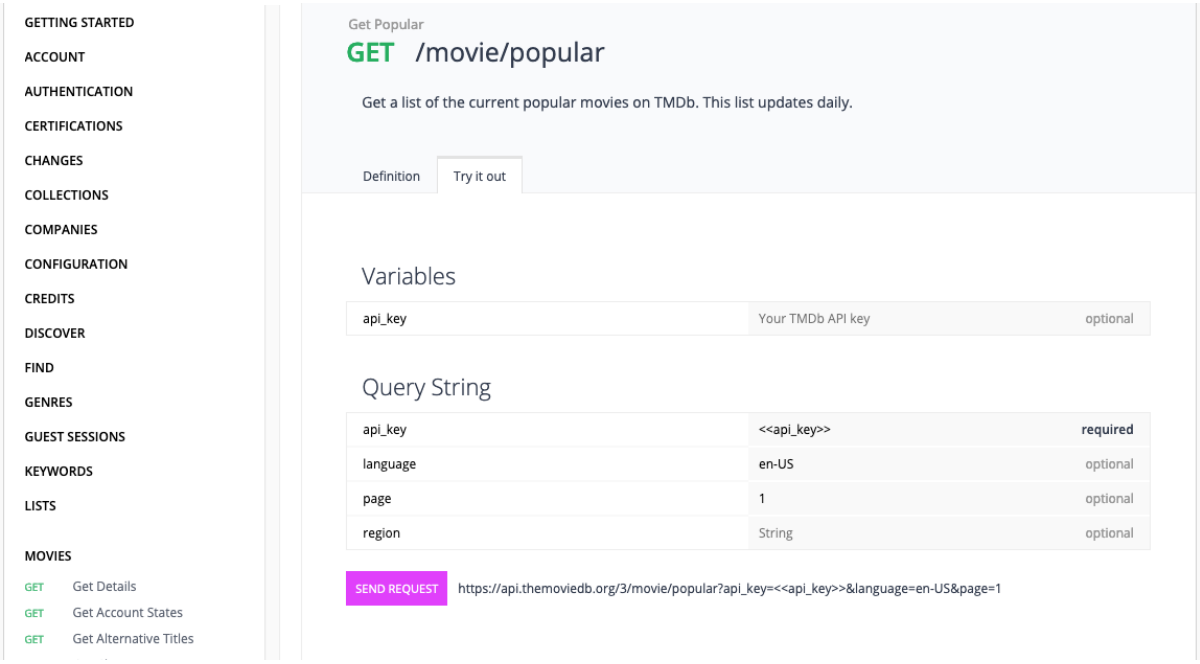
b. https://stackoverflow.com/questions/59915827/how-to-add-a-library-to-flutter-dependencies

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.



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:

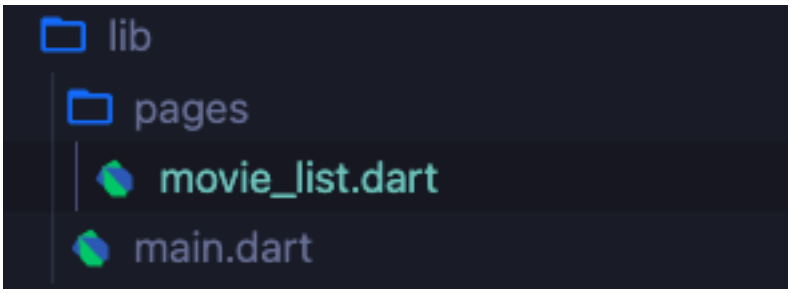
1. Untuk mengakses internet kita harus menambahkan permission internet pada android manifest cari lah file android manifest.xml pada folder android/app/scr/main/AndroidManifest.xml kemudian tambahkan permission internet.

```
<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
      <!-- 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
```

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

```dart
import 'package:flutter/material.dart';

Run | Debug
void main() {
  runApp(MyApp());
}

class MyApp extends StatelessWidget {
  // This widget is the root of your application.
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Flutter Demo',
      theme: ThemeData(
        primarySwatch: Colors.blue,
      ), // ThemeData
      home: MyHomePage(),
    ); // MaterialApp
  }
}

class MyHomePage extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Container();
  }
}
```

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

```dart
import 'package:flutter/material.dart';

class MovieList extends StatefulWidget {
  @override
  _MovieListState createState() => _MovieListState();
}

class _MovieListState extends State<MovieList> {
  @override
  Widget build(BuildContext context) {
    return Container();
  }
}
```

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

```dart
class MyHomePage extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MovieList();
  }
}
```

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.

```dart
import 'package:http/http.dart' as http;

class HttpService {
  final String apiKey = 'isi dengan api key anda';
  final String baseUrl = 'https://api.themoviedb.org/3/movie/popular?api_key=';
}
```

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

```dart
import 'dart:io';
import 'package:http/http.dart' as http;

class HttpService {
  final String apiKey = 'isi dengan api key anda';
  final String baseUrl = 'https://api.themoviedb.org/3/movie/popular?api_key=';

  Future<String> getPopularMovies() async {
    final String uri = baseUrl + apiKey;

    http.Response result = await http.get(Uri.parse(uri));
    if (result.statusCode == HttpStatus.ok) {
      print("Sukses");
      String response = result.body;
      return response;
    } else {
      print("Fail");
      return null;
    }
  }
}
```

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.

```dart
class _MovieListState extends State<MovieList> {
  String result = "";
  HttpService service;
```

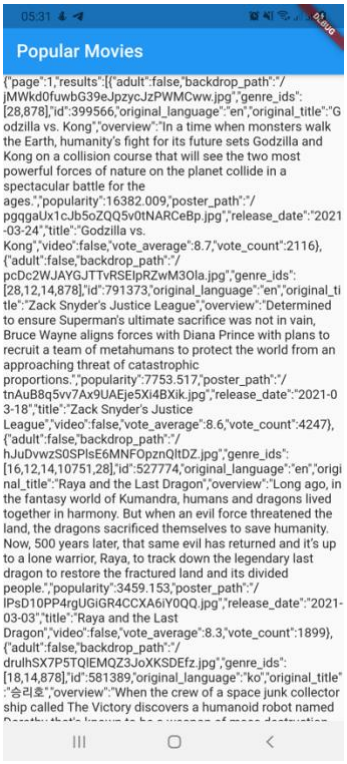
11. Kemudian tambahkan method override init state agar permintaan ke rest api dapat dilakukan ketika state di inisialisasi.

```dart
class _MovieListState extends State<MovieList> {
  String result = "";
  HttpService service;

  @override
  void initState() {
    service = HttpService();
    super.initState();
  }
```

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

```dart
@override
Widget build(BuildContext context) {
  service.getPopularMovies().then((value) => {
        setState(() {
          result = value;
        })
      });
  return Scaffold(
    appBar: AppBar(
      title: Text("Popular Movies"),
    ), // AppBar
    body: Container(
      child: Text(result),
    ), // Container
  ); // Scaffold
}
}
```

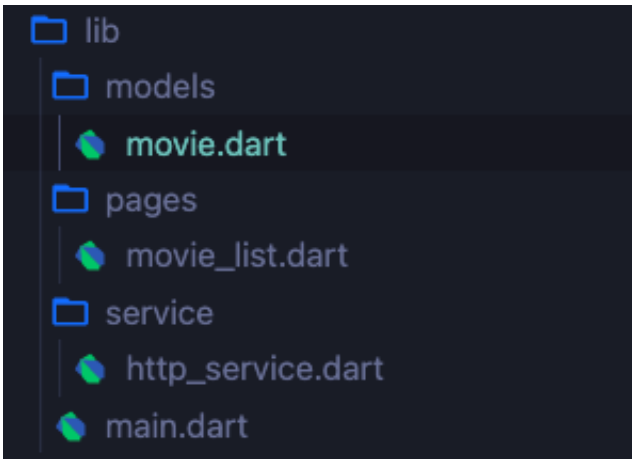
13. Jalankan aplikasi, jika di awal terjadi error reload menggunakan hot restart.



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 construktor seperti dibawah ini.

```
lib > models > movie.dart > ...
class Movie {
  int id;
  String title;
  double voteAverage;
  String overview;
  String posterPath;

  Movie(this.id, this.title, this.voteAverage, this.overview, this.posterPath);
}
```

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

```
lib > models > movie.dart > ...
class Movie {
  int id;
  String title;
  double voteAverage;
  String overview;
  String posterPath;

  Movie(this.id, this.title, this.voteAverage, this.overview, this.posterPath);

  Movie.fromJson(Map<String, dynamic> parsedJson) {
    this.id = parsedJson['id'];
    this.title = parsedJson['title'];
    this.voteAverage = parsedJson['vote_average'] * 1.0;
    this.overview = parsedJson['overview'];
    this.posterPath = parsedJson['poster_path'];
  }
}
```

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

```dart
import 'dart:convert';
import 'dart:io';
import 'package:http/http.dart' as http;
import 'package:moviedb2/models/movie.dart';

class HttpService {
  final String apiKey = 'api key';
  final String baseUrl = 'https://api.themoviedb.org/3/movie/popular?api_key=';

  Future<List> getPopularMovies() async {
    final String uri = baseUrl + apiKey;

    http.Response result = await http.get(Uri.parse(uri));
    if (result.statusCode == HttpStatus.ok) {
      print("Sukses");
      final jsonResponse = json.decode(result.body);
      final moviesMap = jsonResponse['results'];
      List movies = moviesMap.map((i) => Movie.fromJson(i)).toList();
      return movies;
    } else {
      print("Fail");
      return null;
    }
  }
}
```

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

```dart
class _MovieListState extends State<MovieList> {
  int moviesCount;
  List movies;
  HttpService service;
```

2. Tambahkan method initialize() pada class movieliststate

```
class _MovieListState extends State<MovieList> {
  int moviesCount;
  List movies;
  HttpService service;

  Future initialize() async {
    movies = [];
    movies = await service.getPopularMovies();
    setState(() {
      moviesCount = movies.length;
      movies = movies;
    });
  }

```

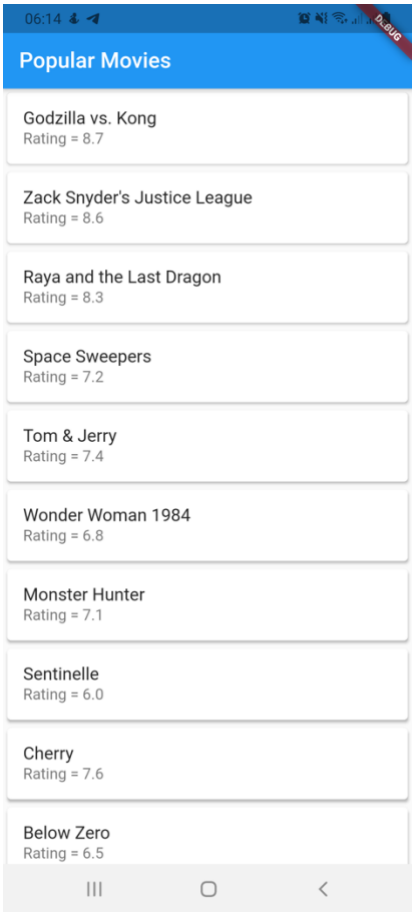
3. Tambahkan function initialize pada initState

```
class _MovieListState extends State<MovieList> {
  int moviesCount;
  List movies;
  HttpService service;

  Future initialize() async {
    movies = [];
    movies = await service.getPopularMovies();
    setState(() {
      moviesCount = movies.length;
      movies = movies;
    });
  }

  @override
  void initState() {
    service = HttpService();
    initialize();
    super.initState();
  }
```

4. Selanjutnya buat listview dan card berdasarkan data dari initialize

```dart
@override
Widget build(BuildContext context) {
  return Scaffold(
      appBar: AppBar(
        title: Text("Popular Movies"),
      ), // AppBar
      body: ListView.builder(
        itemCount: (this.moviesCount == null) ? 0 : this.moviesCount,
        itemBuilder: (context, int position) {
          return Card(
            color: Colors.white,
            elevation: 2.0,
            child: ListTile(
              title: Text(movies[position].title),
              subtitle: Text(
                'Rating = ' + movies[position].voteAverage.toString(),
              ), // Text
            ), // ListTile
          ); // Card
        },
      )); // ListView.builder // Scaffold
}
```

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.

```
return Card(
  color: Colors.white,
  elevation: 2.0,
  child: ListTile(
    title: Text(movies[position].title),
    subtitle: Text(
      'Rating = ' + movies[position].voteAverage.toString(),
    ), // Text
    onTap: () {
      MaterialPageRoute route = MaterialPageRoute(
          builder: (_) => MovieDetail(movies[position])); // MaterialPageRoute
      Navigator.push(context, route);
    },
  ), // ListTile
); // Card
```

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

lib > pages > movie_detail.dart > ...

```
import 'package:flutter/material.dart';

class MovieDetail extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Container();
  }
}
```

3. Lengkapi Movie Detail untuk menerima parameter Movies

```
lib > pages > movie_detail.dart > ...
import 'package:flutter/material.dart';
import 'package:moviedb2/models/movie.dart';

class MovieDetail extends StatelessWidget {
  final Movie movie;
  final String imgPath = 'https://image.tmdb.org/t/p/w500/';

  MovieDetail(this.movie);

  @override
  Widget build(BuildContext context) {
    return Container();
  }
}
```

4. Lengkapi detail widget.

```
@override
Widget build(BuildContext context) {
  String path;
  if (movie.posterPath != null) {
    path = imgPath + movie.posterPath;
  } else {
    path =
        'https://images.freeimages.com/images/large-previews/5eb/movie-clapboard-1184339.jpg';
  }
  double height = MediaQuery.of(context).size.height;
  return Scaffold(
    appBar: AppBar(
      title: Text(movie.title),
    ), // AppBar
    body: SingleChildScrollView(
      child: Center(
        child: Column(
          children: [
            Container(
                padding: EdgeInsets.all(16),
                height: height / 1.5,
                child: Image.network(path)), // Container
            Container(
              child: Text(movie.overview),
              padding: EdgeInsets.only(left: 16, right: 16),
            ), // Container
          ],
        ), // Column
      ), // Center
    ), // SingleChildScrollView
  ); // Scaffold
}
```

5. Challenge Modifikasilah Tampilan agara terlihat lebih menarik