import 'package:flutter/material.dart';

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

import 'dart:convert';

import 'package:cached\_network\_image/cached\_network\_image.dart';

void main() {

runApp(MyApp());

}

class MyApp extends StatelessWidget {

@override

Widget build(BuildContext context) {

return MaterialApp(

home: SplashScreen(),

);

}

}

class SplashScreen extends StatelessWidget {

@override

Widget build(BuildContext context) {

return Scaffold(

body: Center(

child: Image.asset('assets/splash\_image.jpg'),

),

);

}

}

class HomeScreen extends StatefulWidget {

@override

\_HomeScreenState createState() => \_HomeScreenState();

}

class \_HomeScreenState extends State<HomeScreen> {

List<dynamic> movies = [];

bool isLoading = false;

@override

void initState() {

super.initState();

fetchMovies();

}

Future<void> fetchMovies() async {

try {

setState(() {

isLoading = true;

});

final response = await http.get('https://api.tvmaze.com/search/shows?q=all');

if (response.statusCode == 200) {

setState(() {

movies = json.decode(response.body);

});

} else {

throw Exception('Failed to load movies. Status Code: ${response.statusCode}');

}

} catch (error) {

print('Error fetching movies: $error');

} finally {

setState(() {

isLoading = false;

});

}

}

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(

title: Text('Movies'),

actions: [

IconButton(

icon: Icon(Icons.search),

onPressed: () {

Navigator.push(

context,

MaterialPageRoute(builder: (context) => SearchScreen()),

);

},

),

],

),

body: isLoading

? Center(child: CircularProgressIndicator())

: ListView.builder(

itemCount: movies.length,

itemBuilder: (context, index) {

return MovieListItem(movie: movies[index]);

},

),

);

}

}

class MovieListItem extends StatelessWidget {

final dynamic movie;

MovieListItem({required this.movie});

@override

Widget build(BuildContext context) {

return ListTile(

onTap: () {

Navigator.push(

context,

MaterialPageRoute(

builder: (context) => DetailsScreen(movie: movie),

),

);

},

leading: CachedNetworkImage(

imageUrl: movie['show']['image']['medium'],

placeholder: (context, url) => CircularProgressIndicator(),

errorWidget: (context, url, error) => Icon(Icons.error),

),

title: Text(movie['show']['name']),

subtitle: Text(movie['show']['summary']),

);

}

}

class SearchScreen extends StatefulWidget {

@override

\_SearchScreenState createState() => \_SearchScreenState();

}

class \_SearchScreenState extends State<SearchScreen> {

TextEditingController \_searchController = TextEditingController();

List<dynamic> searchResults = [];

bool isLoading = false;

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(

title: TextField(

controller: \_searchController,

decoration: InputDecoration(

hintText: 'Search movies...',

),

),

actions: [

IconButton(

icon: Icon(Icons.search),

onPressed: () {

searchMovies(\_searchController.text);

},

),

],

),

body: isLoading

? Center(child: CircularProgressIndicator())

: ListView.builder(

itemCount: searchResults.length,

itemBuilder: (context, index) {

return MovieListItem(movie: searchResults[index]);

},

),

);

}

Future<void> searchMovies(String searchTerm) async {

try {

setState(() {

isLoading = true;

});

final response =

await http.get('https://api.tvmaze.com/search/shows?q=$searchTerm');

if (response.statusCode == 200) {

setState(() {

searchResults = json.decode(response.body);

});

} else {

throw Exception('Failed to search movies. Status Code: ${response.statusCode}');

}

} catch (error) {

print('Error searching movies: $error');

} finally {

setState(() {

isLoading = false;

});

}

}

}

class DetailsScreen extends StatelessWidget {

final dynamic movie;

DetailsScreen({required this.movie});

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(

title: Text(movie['show']['name']),

),

body: Padding(

padding: const EdgeInsets.all(16.0),

child: Column(

crossAxisAlignment: CrossAxisAlignment.start,

children: [

CachedNetworkImage(

imageUrl: movie['show']['image']['original'],

placeholder: (context, url) => CircularProgressIndicator(),

errorWidget: (context, url, error) => Icon(Icons.error),

),

SizedBox(height: 16),

Text(

'Summary:',

style: TextStyle(

fontSize: 18,

fontWeight: FontWeight.bold,

),

),

Text(

movie['show']['summary'],

style: TextStyle(fontSize: 16),

),

],

),

),

);

}

}

Pubsep.yaml:-

dependencies:

flutter:

sdk: flutter

http: ^0.13.3

cached\_network\_image: ^3.1.0