need flutter dart code generation for the following information with seperate main.dart, helpscreen.dart, homepagescreen.dart, other api resources : Task: Weather Apps app Description: => It is simple app with 2 screen (Help Screen and Homepage Screen). => On first launch, app should show help screen as splash screen (which has heading text/image with skip button; See below description) => In homepage app should weather (See below description)  on help screen should have: => Heading text view: “We show weather for you” => Must include button named ‘Skip’ => Must have any background image frame (Eg: https://www.vhv.rs/dpng/d/427-4270068\_gold-retro decorative-frame-png-free-download-transparent.png ) or any other frame from internet [download image and use it] your content must not collide with frame border. => If the skip button was clicked app should go to homepage directly on later launch. => If user does not click any button, it should redirect automatically to homepage after 5 seconds. Homepage screen should have: => Should contain textbox to enter location name (initially blank) and Search button. => Show temperature view (use widget you see appropriate) in appropriate way to show temperature received from API. => When Save button is clicked, if location name is empty or cannot get temperature from API display error message else update temperature view. => In top bar there should be button to open Help Screen (Note: Help Screen has similar behavior as mentioned above) on temperature view must contain: => temperature in Celsius ( Eg: "temp\_c": 8.0 in below sample response) => temperature in text (Eg: "text": "Clear" in below sample response) => Icon (Eg: "icon": "//cdn.weatherapi.com/weather/64x64/night/113.png" in below sample response) for api and other note are: Note: - - - - - - - - API : https://api.weatherapi.com/v1/current.json?key=API\_KEY&q=London Sign up here and login to get api key: Signup - WeatherAPI.com Use state management tool (GetX or Provider) Follow folder architecture: MVCs Make beautiful UI (UI should be responsive and adaptive) Use text themes Make a button to toggle between light and dark theme Feel free to add more things if you like

**ChatGPT said:**

ChatGPT

Here's the Flutter project structure and code for your weather app as described. The project will use **GetX** for state management and navigation and follow the **MVC pattern**. Below is the breakdown:

**Folder Structure**

css

Copy code

lib/

├── main.dart

├── controllers/

│ ├── weather\_controller.dart

├── models/

│ ├── weather\_model.dart

├── services/

│ ├── weather\_api\_service.dart

├── views/

│ ├── help\_screen.dart

│ ├── home\_screen.dart

├── themes/

│ ├── app\_themes.dart

└── widgets/

├── temperature\_view.dart

**main.dart**

Handles the app's entry point, theme toggling, and routing.

dart

Copy code

import 'package:flutter/material.dart';

import 'package:get/get.dart';

import 'views/help\_screen.dart';

import 'views/home\_screen.dart';

import 'themes/app\_themes.dart';

void main() {

runApp(MyApp());

}

class MyApp extends StatelessWidget {

@override

Widget build(BuildContext context) {

return GetMaterialApp(

title: 'Weather App',

theme: AppThemes.lightTheme,

darkTheme: AppThemes.darkTheme,

themeMode: ThemeMode.system,

debugShowCheckedModeBanner: false,

initialRoute: '/help',

getPages: [

GetPage(name: '/help', page: () => HelpScreen()),

GetPage(name: '/home', page: () => HomeScreen()),

],

);

}

}

**help\_screen.dart**

Help screen that acts as a splash screen.

dart

Copy code

import 'package:flutter/material.dart';

import 'package:get/get.dart';

import 'dart:async';

class HelpScreen extends StatelessWidget {

HelpScreen() {

Timer(Duration(seconds: 5), () {

if (!Get.isSnackbarOpen) {

Get.offNamed('/home');

}

});

}

@override

Widget build(BuildContext context) {

return Scaffold(

body: Stack(

fit: StackFit.expand,

children: [

Image.network(

'https://www.vhv.rs/dpng/d/427-4270068\_gold-retro-decorative-frame-png-free-download-transparent.png',

fit: BoxFit.cover,

),

Center(

child: Column(

mainAxisAlignment: MainAxisAlignment.center,

children: [

Text(

'We show weather for you',

style: Theme.of(context).textTheme.headline4,

textAlign: TextAlign.center,

),

SizedBox(height: 20),

ElevatedButton(

onPressed: () => Get.offNamed('/home'),

child: Text('Skip'),

),

],

),

),

],

),

);

}

}

**home\_screen.dart**

Home screen for searching and displaying weather data.

dart

Copy code

import 'package:flutter/material.dart';

import 'package:get/get.dart';

import '../controllers/weather\_controller.dart';

import '../widgets/temperature\_view.dart';

class HomeScreen extends StatelessWidget {

final WeatherController weatherController = Get.put(WeatherController());

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(

title: Text('Weather App'),

actions: [

IconButton(

icon: Icon(Icons.help\_outline),

onPressed: () => Get.toNamed('/help'),

),

IconButton(

icon: Icon(Icons.brightness\_6),

onPressed: () => weatherController.toggleTheme(),

),

],

),

body: Padding(

padding: const EdgeInsets.all(16.0),

child: Column(

children: [

TextField(

controller: weatherController.locationController,

decoration: InputDecoration(

labelText: 'Enter location',

border: OutlineInputBorder(),

),

),

SizedBox(height: 10),

ElevatedButton(

onPressed: weatherController.fetchWeather,

child: Text('Search'),

),

SizedBox(height: 20),

Obx(() {

if (weatherController.weather == null) {

return Text('Enter a location to get weather details.');

} else if (weatherController.error != null) {

return Text('Error: ${weatherController.error}');

}

return TemperatureView(weatherController.weather!);

}),

],

),

),

);

}

}

**weather\_controller.dart**

Controller for managing state and API interaction.

dart

Copy code

import 'package:flutter/material.dart';

import 'package:get/get.dart';

import '../models/weather\_model.dart';

import '../services/weather\_api\_service.dart';

class WeatherController extends GetxController {

var weather = Rxn<WeatherModel>();

var error = RxnString();

var isDarkMode = false.obs;

final locationController = TextEditingController();

void fetchWeather() async {

final location = locationController.text;

if (location.isEmpty) {

error.value = 'Location cannot be empty.';

weather.value = null;

return;

}

try {

weather.value = await WeatherApiService.fetchWeather(location);

error.value = null;

} catch (e) {

error.value = e.toString();

weather.value = null;

}

}

void toggleTheme() {

isDarkMode.value = !isDarkMode.value;

Get.changeThemeMode(isDarkMode.value ? ThemeMode.dark : ThemeMode.light);

}

}

**weather\_api\_service.dart**

Service for fetching weather data.

dart

Copy code

import 'dart:convert';

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

import '../models/weather\_model.dart';

class WeatherApiService {

static const String apiKey = 'YOUR\_API\_KEY';

static const String baseUrl = 'https://api.weatherapi.com/v1/current.json';

static Future<WeatherModel> fetchWeather(String location) async {

final response = await http.get(Uri.parse('$baseUrl?key=$apiKey&q=$location'));

if (response.statusCode == 200) {

return WeatherModel.fromJson(json.decode(response.body));

} else {

throw Exception('Failed to load weather data.');

}

}

}

**weather\_model.dart**

Model to parse API response.

dart

Copy code

class WeatherModel {

final double tempC;

final String text;

final String icon;

WeatherModel({required this.tempC, required this.text, required this.icon});

factory WeatherModel.fromJson(Map<String, dynamic> json) {

return WeatherModel(

tempC: json['current']['temp\_c'],

text: json['current']['condition']['text'],

icon: json['current']['condition']['icon'],

);

}

}

**temperature\_view.dart**

Widget to display weather information.

dart

Copy code

import 'package:flutter/material.dart';

import '../models/weather\_model.dart';

class TemperatureView extends StatelessWidget {

final WeatherModel weather;

const TemperatureView(this.weather);

@override

Widget build(BuildContext context) {

return Column(

mainAxisSize: MainAxisSize.min,

children: [

Text(

'${weather.tempC}°C',

style: Theme.of(context).textTheme.headline2,

),

Text(

weather.text,

style: Theme.of(context).textTheme.headline6,

),

Image.network('https:${weather.icon}'),

],

);

}

}

**app\_themes.dart**

Themes for the app.

dart

Copy code

import 'package:flutter/material.dart';

class AppThemes {

static final lightTheme = ThemeData(

brightness: Brightness.light,

primarySwatch: Colors.blue,

);

static final darkTheme = ThemeData(

brightness: Brightness.dark,

primarySwatch: Colors.blue,

);

}

Replace YOUR\_API\_KEY with your actual API key from WeatherAPI. This setup provides clean separation of concerns, uses GetX for state management, and follows the MVC pattern. Let me know if you need further adjustments!

RUNNING CODE will generate following error:  
  
this all code show folowing error:

1. on help\_screen.dart: on line 30: The getter 'headline4' isn't defined for the type 'TextTheme'.

2. on home\_screen.dart on line 48: The argument type 'Rxn<WeatherModel>' can't be assigned to the parameter type 'WeatherModel'.

And,

3. on temperature\_view.dart: on line 18: The getter 'headline2' isn't defined for the type 'TextTheme'. and on line 22: The getter 'headline6' isn't defined for the type 'TextTheme'. please adress this issue and fix in above code again.

New update code are:

he errors you are encountering are due to changes in the Flutter framework and Dart SDK over time. Specifically:

1. **headline4, headline2, and headline6 are now replaced** with displayLarge, displayMedium, and titleLarge respectively in the newer versions of Flutter.
2. **Rxn<T> incompatibility**: This occurs because Rxn (a reactive nullable variable) isn't directly compatible with non-reactive types like WeatherModel. Instead, we need to use .value to access the underlying data.

Here’s the updated code addressing these issues:

**Updated help\_screen.dart**

dart

Copy code

import 'package:flutter/material.dart';

import 'package:get/get.dart';

import 'dart:async';

class HelpScreen extends StatelessWidget {

HelpScreen() {

Timer(Duration(seconds: 5), () {

if (!Get.isSnackbarOpen) {

Get.offNamed('/home');

}

});

}

@override

Widget build(BuildContext context) {

return Scaffold(

body: Stack(

fit: StackFit.expand,

children: [

Image.network(

'https://www.vhv.rs/dpng/d/427-4270068\_gold-retro-decorative-frame-png-free-download-transparent.png',

fit: BoxFit.cover,

),

Center(

child: Column(

mainAxisAlignment: MainAxisAlignment.center,

children: [

Text(

'We show weather for you',

style: Theme.of(context).textTheme.displayMedium,

textAlign: TextAlign.center,

),

SizedBox(height: 20),

ElevatedButton(

onPressed: () => Get.offNamed('/home'),

child: Text('Skip'),

),

],

),

),

],

),

);

}

}

**Updated home\_screen.dart**

dart

Copy code

import 'package:flutter/material.dart';

import 'package:get/get.dart';

import '../controllers/weather\_controller.dart';

import '../widgets/temperature\_view.dart';

class HomeScreen extends StatelessWidget {

final WeatherController weatherController = Get.put(WeatherController());

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(

title: Text('Weather App'),

actions: [

IconButton(

icon: Icon(Icons.help\_outline),

onPressed: () => Get.toNamed('/help'),

),

IconButton(

icon: Icon(Icons.brightness\_6),

onPressed: () => weatherController.toggleTheme(),

),

],

),

body: Padding(

padding: const EdgeInsets.all(16.0),

child: Column(

children: [

TextField(

controller: weatherController.locationController,

decoration: InputDecoration(

labelText: 'Enter location',

border: OutlineInputBorder(),

),

),

SizedBox(height: 10),

ElevatedButton(

onPressed: weatherController.fetchWeather,

child: Text('Search'),

),

SizedBox(height: 20),

Obx(() {

if (weatherController.weather.value == null) {

return Text('Enter a location to get weather details.');

} else if (weatherController.error.value != null) {

return Text('Error: ${weatherController.error.value}');

}

return TemperatureView(weatherController.weather.value!);

}),

],

),

),

);

}

}

**Updated temperature\_view.dart**

dart

Copy code

import 'package:flutter/material.dart';

import '../models/weather\_model.dart';

class TemperatureView extends StatelessWidget {

final WeatherModel weather;

const TemperatureView(this.weather);

@override

Widget build(BuildContext context) {

return Column(

mainAxisSize: MainAxisSize.min,

children: [

Text(

'${weather.tempC}°C',

style: Theme.of(context).textTheme.displayLarge,

),

Text(

weather.text,

style: Theme.of(context).textTheme.titleLarge,

),

Image.network('https:${weather.icon}'),

],

);

}

}

**Explanation of Fixes**

1. **Updated TextTheme getters**:
   * headline4 → displayMedium
   * headline2 → displayLarge
   * headline6 → titleLarge These changes align with Flutter’s newer typography system.
2. **Fixed Rxn<WeatherModel> issue**:
   * Used .value to access the reactive value of Rxn<WeatherModel> in the Obx widget.
   * Added a null check (.value!) before passing the value to TemperatureView.

Let me know if any other issues arise!

succesfully running app with location enter text bod and search button. after putting Nepal in text box and click search, this will not sown any data. what location should i choose here and if data not feating, need show message with processing or no data found!

**ChatGPT said:**

ChatGPT

For the WeatherAPI to return valid weather data, the location entered in the text box must match a recognized location by the API. For example, valid inputs include:

1. City names (e.g., **Kathmandu**, **London**, **Tokyo**).
2. Country names (e.g., **Nepal**, **India**).
3. Latitude and Longitude (e.g., **27.7172,85.3240** for Kathmandu).
4. ZIP/postal codes (if applicable for certain countries).

**Update to Handle Missing Data or Processing States**

To enhance the app functionality:

1. Show a **loading indicator** while fetching data.
2. Display a **"No data found"** message if the API doesn't return data or the location is invalid.

Here’s how to implement these changes:

**Updated weather\_controller.dart**

dart

Copy code

import 'package:flutter/material.dart';

import 'package:get/get.dart';

import '../models/weather\_model.dart';

import '../services/weather\_api\_service.dart';

class WeatherController extends GetxController {

var weather = Rxn<WeatherModel>();

var error = RxnString();

var isLoading = false.obs; // Add loading state

final locationController = TextEditingController();

void fetchWeather() async {

final location = locationController.text.trim();

if (location.isEmpty) {

error.value = 'Location cannot be empty.';

weather.value = null;

return;

}

isLoading.value = true; // Set loading state to true

error.value = null;

try {

final fetchedWeather = await WeatherApiService.fetchWeather(location);

weather.value = fetchedWeather;

if (fetchedWeather == null) {

error.value = 'No data found for the location "$location".';

}

} catch (e) {

error.value = 'Error fetching weather: ${e.toString()}';

} finally {

isLoading.value = false; // Reset loading state

}

}

}

**Updated home\_screen.dart**

Update the UI to handle the new loading and error states.

dart

Copy code

import 'package:flutter/material.dart';

import 'package:get/get.dart';

import '../controllers/weather\_controller.dart';

import '../widgets/temperature\_view.dart';

class HomeScreen extends StatelessWidget {

final WeatherController weatherController = Get.put(WeatherController());

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(

title: Text('Weather App'),

actions: [

IconButton(

icon: Icon(Icons.help\_outline),

onPressed: () => Get.toNamed('/help'),

),

IconButton(

icon: Icon(Icons.brightness\_6),

onPressed: () => weatherController.toggleTheme(),

),

],

),

body: Padding(

padding: const EdgeInsets.all(16.0),

child: Column(

children: [

TextField(

controller: weatherController.locationController,

decoration: InputDecoration(

labelText: 'Enter location',

border: OutlineInputBorder(),

),

),

SizedBox(height: 10),

ElevatedButton(

onPressed: weatherController.fetchWeather,

child: Text('Search'),

),

SizedBox(height: 20),

Obx(() {

if (weatherController.isLoading.value) {

return CircularProgressIndicator(); // Show loading indicator

}

if (weatherController.error.value != null) {

return Text(

weatherController.error.value!,

style: TextStyle(color: Colors.red),

);

}

if (weatherController.weather.value == null) {

return Text('Enter a location to get weather details.');

}

return TemperatureView(weatherController.weather.value!);

}),

],

),

),

);

}

}

**Changes Made**

1. **Loading Indicator**:
   * isLoading state is used to display a circular progress indicator while waiting for the API response.
2. **Error Handling**:
   * Displays a descriptive error message (e.g., "No data found for the location").
3. **Data Validity**:
   * If no weather data is returned, an error message is shown instead of an empty view.

**Example Input for Testing**

Try these inputs for valid results:

* **Kathmandu**
* **Pokhara**
* **New York**
* **London**

Invalid inputs (e.g., gibberish text like **xyz123**) will now show an appropriate error message.

Let me know if further enhancements are needed! 🚀