**Flutter App Development Course**

**DAY # 7**

**Task 1:**

**Source Code:**

**Main.dart**

import 'package:flutter/material.dart';

import 'package:clima/screens/loading\_screen.dart';

void main() => runApp(MyApp());

class MyApp extends StatelessWidget {

@override

Widget build(BuildContext context) {

return MaterialApp(

debugShowCheckedModeBanner: false,

theme: ThemeData.dark(),

home: LoadingScreen(),

);

}

}

**city\_screen.dart**

import 'package:flutter/material.dart';

import 'package:clima/utilities/constants.dart';

import 'package:font\_awesome\_flutter/font\_awesome\_flutter.dart';

class CityScreen extends StatefulWidget {

@override

\_CityScreenState createState() => \_CityScreenState();

}

class \_CityScreenState extends State<CityScreen> {

String cityName;

@override

Widget build(BuildContext context) {

return Scaffold(

body: Container(

decoration: BoxDecoration(

image: DecorationImage(

image: AssetImage('images/earth.jpg'),

fit: BoxFit.cover,

),

),

constraints: BoxConstraints.expand(),

child: SafeArea(

child: Column(

children: <Widget>[

SizedBox(height: 20,),

Container(

padding: EdgeInsets.all(30.0),

child: TextField(

style: TextStyle(

color: Colors.cyan,

),

decoration: kTextFieldInputDecoration,

onChanged: (value) {

cityName = value;

},

),

),

TextButton(

onPressed: () {

Navigator.pop(context, cityName);

},

child: Text(

'Get Weather',

style: kButtonTextStyle,

),

),

Align(

alignment: Alignment.center,

child: TextButton(

onPressed: () {

Navigator.pop(context);

},

child: Row( mainAxisAlignment: MainAxisAlignment.center,

children: [

Icon(

FontAwesomeIcons.angleLeft,

size: 30.0,color: Colors.grey[700],

),

Text("Back" , style: TextStyle(

fontSize: 25,

color: Colors.grey[700],

fontWeight: FontWeight.bold),),

],

),

),

),

],

),

),

),

);

}

}

**Loading\_screen.dart**

import 'package:flutter/material.dart';

import 'location\_screen.dart';

import 'package:flutter\_spinkit/flutter\_spinkit.dart';

import 'package:clima/services/weather.dart';

class LoadingScreen extends StatefulWidget {

@override

State<StatefulWidget> createState() {

return \_LoadingScreenState();

}

}

class \_LoadingScreenState extends State<LoadingScreen> {

@override

void initState() {

super.initState();

getLocationData();

}

void getLocationData() async {

var weatherData = await WeatherModel().getLocationWeather();

Navigator.push(context, MaterialPageRoute(builder: (context) {

return LocationScreen(

locationWeather: weatherData,

);

}));

}

@override

Widget build(BuildContext context) {

return Scaffold(

body: Center(

child: SpinKitSpinningLines(color: Colors.cyan , duration: Duration(seconds: 2),size: 120.0,)

),

);

}

}

**Location\_screen.dart**

import 'package:flutter/material.dart';

import 'package:clima/utilities/constants.dart';

import 'package:clima/services/weather.dart';

import 'package:font\_awesome\_flutter/font\_awesome\_flutter.dart';

import 'city\_screen.dart';

import 'package:intl/intl.dart';

class LocationScreen extends StatefulWidget {

LocationScreen({this.locationWeather});

final locationWeather;

@override

\_LocationScreenState createState() => \_LocationScreenState();

}

class \_LocationScreenState extends State<LocationScreen> {

WeatherModel weather = WeatherModel();

int temperature;

String weatherIcon;

String cityName;

String weatherMessage;

String weatherDescription;

String country;

// String now = DateFormat("yyyy-MM-dd").format

@override

void initState() {

super.initState();

updateUI(widget.locationWeather);

}

void updateUI(dynamic weatherData) {

setState(() {

if (weatherData == null) {

temperature = 0;

weatherIcon = 'Error';

weatherMessage = 'Unable to get weather data';

cityName = '';

return;

}

double temp = weatherData["main"]["temp"];

temperature = temp.toInt() ;

print("Temperature : $temperature");

var condition = weatherData["weather"][0]["id"];

print("City ID : $condition");

// longitude = weatherData["coord"]["lon"];

// print("City Longitude : $longitude");

weatherDescription = weatherData["weather"][0]["description"];

print("Weather Description : $weatherDescription");

cityName = weatherData["name"];

print("City Name : $cityName");

country = weatherData["sys"]["country"];

print("Country Name : $country");

weatherIcon = weather.getWeatherIcon(condition);

weatherMessage = weather.getMessage(condition);

});

}

@override

Widget build(BuildContext context) {

return Scaffold(

body: Container(

decoration: BoxDecoration(

image: DecorationImage(

image: AssetImage('$weatherMessage'),

fit: BoxFit.cover,

colorFilter: ColorFilter.mode(

Colors.white.withOpacity(0.8), BlendMode.dstATop),

),

),

constraints: BoxConstraints.expand(),

child: SafeArea(

child: Column(

// mainAxisAlignment: MainAxisAlignment.center,

crossAxisAlignment: CrossAxisAlignment.stretch,

children: <Widget>[

Row(

mainAxisAlignment: MainAxisAlignment.spaceBetween,

children: <Widget>[

TextButton(

onPressed: () async {

var weatherData = await weather.getLocationWeather();

updateUI(weatherData);

},

child: Icon(

Icons.near\_me,

size: 40.0,

color: Colors.black

),

),

TextButton(

onPressed: () async {

var typedName = await Navigator.push(

context,

MaterialPageRoute(

builder: (context) {

return CityScreen();

},

),

);

if (typedName != null) {

var weatherData =

await weather.getCityWeather(typedName);

updateUI(weatherData);

}

},

child: Icon(

FontAwesomeIcons.mapMarkedAlt,

size: 40.0,

color: Colors.black,

),

),

],

),

SizedBox(height: MediaQuery.of(context).size.height\*0.03,),

Center(

child: Text(

weatherIcon,

style: kConditionTextStyle,

),

),

Padding(

padding: EdgeInsets.only(left: 15.0),

child: Row(

mainAxisAlignment: MainAxisAlignment.center,

children: <Widget>[

Text(

'$temperature',

style: kTempTextStyle,

),

Padding(

padding: const EdgeInsets.only(left:15.0),

child: Text("°", style:TextStyle(fontSize: 80, textBaseline: TextBaseline.alphabetic),),

),

Padding(

padding: const EdgeInsets.only(bottom:60.0 , left: 5),

child: Text("C", style:TextStyle(fontSize: 60, textBaseline: TextBaseline.alphabetic),),

),

],

),

),

Row( mainAxisAlignment: MainAxisAlignment.center,

children: [

Center(

child: Text(

'$cityName , ',

textAlign: TextAlign.right,

style: TextStyle(fontSize: 25, color: Colors.white),

),

),

Center(

child: Text(

'$country',

textAlign: TextAlign.right,

style: TextStyle(fontSize: 25, color: Colors.white),

),

),

],

),

SizedBox(height: 5,),

Center(

child: Text(

'$weatherDescription'.toUpperCase(),

textAlign: TextAlign.right,

style: TextStyle(fontSize: 25, color: Colors.white, fontFamily: "Spartan MB"),

),

),

Padding(

padding: EdgeInsets.only(right: 15.0,top: 140),

child: Center(child:

Text(DateFormat("dd-MM-yyyy").format(DateTime.now()),style: TextStyle(color: Colors.grey ,letterSpacing: 3,fontSize: 20),)

),

),

// Text("$now"),

],

),

),

),

);

}

}**.**

**Location.dart**

import 'package:geolocator/geolocator.dart';

class Location {

double latitude;

double longitude;

Future<void> getCurrentLocation() async {

try {

Position position = await Geolocator().getCurrentPosition(desiredAccuracy: LocationAccuracy.low);

// LocationPermission permission = await Geolocator.requestPermission();

// print(position);

// print(permission);

// LocationPermission pmission = await Geolocator.checkPermission();

// await Geolocator.openAppSettings();

// await Geolocator.openLocationSettings();

latitude = position.latitude;

longitude = position.longitude;

} catch (e) {

print(e);

}

}

}**.**

**Networking.dart**

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

import 'dart:convert';

class NetworkHelper {

NetworkHelper(this.url);

final String url;

Future getData() async {

http.Response response = await http.get(url);

// print(response.body);

// print(response.statusCode);

if (response.statusCode == 200) {

String data = response.body;

print(data);

return jsonDecode(data);

} else {

print(response.statusCode);

}

}

}

**Weather.dart**

import 'package:clima/services/location.dart';

import 'package:clima/services/networking.dart';

const apiKey = 'e5a3df01f4ffe9eb9f1c52b9933af374';

const openWeatherMapURL = 'https://api.openweathermap.org/data/2.5/weather';

class WeatherModel {

Future<dynamic> getCityWeather(String cityName) async {

NetworkHelper networkHelper = NetworkHelper('$openWeatherMapURL?q=$cityName&appid=$apiKey&units=metric');

var weatherData = await networkHelper.getData();

return weatherData;

}

Future<dynamic> getLocationWeather() async {

Location location = Location();

await location.getCurrentLocation();

// latitude = location.latitude;

// longitude = location.longitude;

// print("\*\*\* Latitude : ${latitude}");

// print("\*\*\* Longitude : ${longitude}");

NetworkHelper networkHelper = NetworkHelper('$openWeatherMapURL?lat=${location.latitude}&lon=${location.longitude}&appid=$apiKey&units=metric');

var weatherData = await networkHelper.getData();

return weatherData;

}

String getWeatherIcon(int condition) {

if (condition == 801) {

return '🌩';

} else if (condition == 802) {

return '🌧';

} else if (condition < 600) {

return '☔️';

} else if (condition == 803) {

return '☃️';

} else if (condition < 800) {

return '🌫';

} else if (condition == 800) {

return '☀️';

} else if (condition == 804) {

return '☁️';

} else {

return '🤷‍';

}

}

String getMessage(int condition) {

if (condition == 801) {

return 'images/1.jpg';

} else if (condition == 802) {

return 'images/5.jpg';

} else if (condition < 600) { // rain

return 'images/7.jpg';

} else if (condition == 803) {

return 'images/3.jpg';

} else if (condition < 800) {

return 'images/6.jpg'; // haze

} else if (condition == 800) {

return 'images/2.jpg';

} else if (condition == 804) {

return 'images/4.jpg';

} else {

return 'images/8.jpg';

}

}

}

// if (temp > 25) {

// return 'It\'s 🍦 time';

// } else if (temp > 20) {

// return 'Time for shorts and 👕';

// } else if (temp < 10) {

// return 'You\'ll need 🧣 and 🧤';

// } else {

// return 'Bring a 🧥 just in case';

// }

**constants.dart**

import 'package:flutter/material.dart';

const kTempTextStyle = TextStyle(

fontFamily: 'Spartan MB',

fontSize: 80.0,

);

const kMessageTextStyle = TextStyle(

fontFamily: 'Spartan MB',

fontSize: 30.0,

);

const kButtonTextStyle = TextStyle(

fontSize: 30.0,

fontFamily: 'Spartan MB',

);

const kConditionTextStyle = TextStyle(

fontSize: 90.0,

);

const kTextFieldInputDecoration = InputDecoration(

hintText: "Search Location",

filled: true,

icon: Icon(Icons.search,size: 28,),

);

// InputDecoration(

// filled: true,

// fillColor: Colors.white,

// icon: Icon(

// Icons.location\_city,

// color: Colors.white,

// ),

// hintText: 'Enter City Name',

// hintStyle: TextStyle(

// color: Colors.grey,

// ),

// border: OutlineInputBorder(

// borderRadius: BorderRadius.all(

// Radius.circular(10.0),

// ),

// borderSide: BorderSide.none,

// ),

// );

**Output:**







