**Name:** Wasi Ur Rehman Qamar **Registration Number:** 19-NTU-CS-1130

**LAB 9 Mobile Application Development**

* **Links**
* [**https://github.com/WasiUrRehmanQamar/MobileApplicationDevelopment**](https://github.com/WasiUrRehmanQamar/MobileApplicationDevelopment)
* [**https://lab9-mad.web.app/**](https://lab9-mad.web.app/)
* **Main**

// ignore\_for\_file: prefer\_const\_constructors, unused\_import, deprecated\_member\_use, avoid\_unnecessary\_containers, avoid\_print, prefer\_const\_literals\_to\_create\_immutables, non\_constant\_identifier\_names

import 'package:flutter/material.dart';

import 'package:geolocator/geolocator.dart';

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

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

import 'dart:convert';

import 'package:lab9\_wasi1130/LocationScreen.dart';

void main() {

  runApp(

    MaterialApp(

      home: LoadingScreen(),

    ),

  );

}

dynamic WeatherData;

dynamic Latitude;

dynamic Longitude;

dynamic PermissionStatus;

class NetworkHelper {

  final Uri uri;

  NetworkHelper(this.uri);

  Future getdata() async {

    http.Response res = await http.get(uri);

    if (res.statusCode == 200) {

      String data = res.body;

      return jsonDecode(data);

    } else {

      print(res.statusCode);

    }

  }

}

class LoadingScreen extends StatefulWidget {

  const LoadingScreen({Key? key}) : super(key: key);

  @override

  State<LoadingScreen> createState() => \_MyAppState();

}

class \_MyAppState extends State<LoadingScreen> {

  /\*

    3. Modify the code done in task 1 to print the longitude and latitude

    of the current position on start of app instead of button press.

  \*/

  /\* Start \*/

  @override

  initState() {

    getLocation();

    checklocationStatus();

    super.initState();

  }

  /\* End \*/

  /\*

      1.  Implement the “geolocator” package to determine the current position

      of the device upon click of the button. After determining the position of

      longitude and latitude of the current position.

  \*/

  /\* Start \*/

  void getLocation() async {

    Position position = await Geolocator.getCurrentPosition(

        desiredAccuracy: LocationAccuracy.best);

    print('Location Called');

    print(position.latitude);

    print(position.longitude);

    setState(() {

      Latitude = position.latitude;

      Longitude = position.longitude;

    });

    NetworkHelper helper = NetworkHelper(

      Uri.parse(

          'http://api.openweathermap.org/data/2.5/weather?units=metric&lat=$Latitude&lon=$Longitude&appid=114922bfcf0309eb51a43eec8809aaf4'),

    );

    WeatherData = await helper.getdata();

  }

  /\* End \*/

  /\*

    2.  Determine the current permission status for GPS in the App.

  \*/

  /\* Start \*/

  void checklocationStatus() async {

    LocationPermission Permissions = await Geolocator.checkPermission();

    if (Permissions == LocationPermission.denied) {

      setState(() {

        PermissionStatus = 'Permission Denied';

      });

      print("Permission Denied");

    } else {

      setState(() {

        PermissionStatus = 'Permission Allowed';

      });

      print("Permission Allowed");

    }

  }

  /\* End \*/

  @override

  Widget build(BuildContext context) {

    return Scaffold(

      appBar: AppBar(

        title: Center(

          child: Text('Weather App - Loading Screen'),

        ),

      ),

      body: Center(

        child: Container(

          child: Column(

            mainAxisAlignment: MainAxisAlignment.center,

            crossAxisAlignment: CrossAxisAlignment.center,

            children: [

              Center(

                child: SpinKitChasingDots(

                  color: Colors.grey,

                  size: 40,

                ),

              ),

              SizedBox(

                height: 30,

              ),

              ElevatedButton(

                onPressed: () {

                  getLocation();

                },

                child: Text('Click Here To Get Latitude & Longitude'),

              ),

              SizedBox(

                height: 30,

              ),

              Text('Latitude : $Latitude'),

              SizedBox(

                height: 30,

              ),

              Text('Longitude : $Longitude'),

              SizedBox(

                height: 30,

              ),

              ElevatedButton(

                  onPressed: () {

                    checklocationStatus();

                  },

                  child: Text('Check Permission Status')),

              SizedBox(

                height: 30,

              ),

              Text('Permission Status: $PermissionStatus'),

              SizedBox(

                height: 30,

              ),

              ElevatedButton(

                onPressed: () {

                  Navigator.push(

                    context,

                    MaterialPageRoute(

                        builder: (context) => LocationScreen(

                              Data: WeatherData,

                            )),

                  );

                },

                child: Text('Go to Location Screen'),

              )

            ],

          ),

        ),

      ),

    );

  }

}

* **Location Screen**

// ignore\_for\_file: camel\_case\_types, prefer\_typing\_uninitialized\_variables, override\_on\_non\_overriding\_member, annotate\_overrides, avoid\_print, unused\_import, non\_constant\_identifier\_names, prefer\_const\_constructors, use\_key\_in\_widget\_constructors

import 'package:flutter/material.dart';

import 'dart:convert';

class WeatherIcon {

  Widget getIcon(int condition) {

    if (condition < 300) {

      return Image.network('http://openweathermap.org/img/wn/02d@2x.png');

    } else if (condition < 400) {

      return Image.network('http://openweathermap.org/img/wn/09d@2x.png');

    } else if (condition < 600) {

      return Image.network('http://openweathermap.org/img/wn/10d@2x.png');

    } else if (condition < 700) {

      return Image.network('http://openweathermap.org/img/wn/13d@2x.png');

    } else if (condition < 800) {

      return Image.network('http://openweathermap.org/img/wn/04d@2x.png');

    } else if (condition == 800) {

      return Image.network('http://openweathermap.org/img/wn/01d@2x.png');

    } else if (condition <= 804) {

      return Image.network('http://openweathermap.org/img/wn/03d@2x.png');

    } else {

      return Image.network(

          'https://www.pngfind.com/mpng/iJhbi\_question-mark-png-image-transparent-white-question-mark/');

    }

  }

}

class LocationScreen extends StatefulWidget {

  const LocationScreen({this.Data});

  final dynamic Data;

  @override

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

}

class \_LocationScreenState extends State<LocationScreen> {

  WeatherIcon icons = WeatherIcon();

  dynamic Temperature;

  dynamic City;

  dynamic Icon;

  @override

  void Get(dynamic WeatherData) {

    Temperature = WeatherData['main']['temp'];

    var weathercode = WeatherData['weather'][0]['id'];

    Icon = icons.getIcon(weathercode);

    City = WeatherData['name'].toString();

  }

  void initState() {

    super.initState();

    Get(widget.Data);

  }

  Widget build(BuildContext context) {

    return MaterialApp(

      home: Scaffold(

        appBar: AppBar(

          title: Text('Location Screen'),

        ),

        body: Center(

          child: Column(

            mainAxisAlignment: MainAxisAlignment.center,

            crossAxisAlignment: CrossAxisAlignment.center,

            children: [

              Row(

                mainAxisAlignment: MainAxisAlignment.center,

                crossAxisAlignment: CrossAxisAlignment.center,

                children: [

                  Text('Temperature : '),

                  Text(Temperature.toString()),

                ],

              ),

              SizedBox(

                height: 30,

              ),

              Row(

                mainAxisAlignment: MainAxisAlignment.center,

                crossAxisAlignment: CrossAxisAlignment.center,

                children: [

                  Text('City : '),

                  Text(City.toString()),

                ],

              ),

              Icon,

            ],

          ),

        ),

      ),

    );

  }

}

Graphical user interface, application

Description automatically generated

**Graphical user interface, text, application

Description automatically generated**

**Graphical user interface, text, application

Description automatically generated**