// import 'package:cloud\_firestore/cloud\_firestore.dart';  
// import 'package:flutter/material.dart';  
// import 'package:flutter\_bloc/flutter\_bloc.dart';  
// import 'package:flutter\_screenutil/flutter\_screenutil.dart';  
//  
// import 'Bloc/Image get Bloc/image\_get\_bloc.dart';  
//  
//  
// class Succesfull extends StatefulWidget {  
// final String id;  
//  
// const Succesfull({super.key, required this.id});  
//  
// @override  
// State<Succesfull> createState() => \_SuccesfullState();  
// }  
//  
// class \_SuccesfullState extends State<Succesfull> {  
// final firestore = FirebaseFirestore.instance.collection("Requests");  
//  
// // @override  
// // void initState() {  
// // super.initState();  
// // // Ensuring the event is added after the widget tree is initialized  
// // Future.microtask(() {  
// // BlocProvider.of<ImageGetBloc>(context).add(FeatchGetImage(Id: widget.id));  
// // });  
// //  
// // print('oooooooooooooo${widget.id}');  
// // }  
//  
// // Function to update the Firestore document with the image URL  
// Future<void> updateImageInFirestore(String imageUrl) async {  
// try {  
// // Update the document with the image URL in the 'img' field  
// await firestore.doc(widget.id).update({  
// 'image': imageUrl, // Store the image URL in 'img' field  
// });  
// print('Image URL successfully updated in Firestore.');  
// } catch (e) {  
// print('Failed to update image URL in Firestore: $e');  
// }  
// }  
//  
// @override  
// Widget build(BuildContext context) {  
// return Scaffold(  
// body: Container(  
// width: 1440.w, // Full container width  
// height: 800.h, // Full container height  
// decoration: BoxDecoration(  
// image: DecorationImage(  
// image: AssetImage("assets/img.jpg"), // Background image  
// fit: BoxFit.cover, // Full-screen image  
// ),  
// ),  
// child:  
// Container(  
// color: Colors.grey.withOpacity(0.4),  
// child: Column(  
// mainAxisAlignment: MainAxisAlignment.center,  
// children: [  
// Row(  
// mainAxisAlignment: MainAxisAlignment.center,  
// children: [  
// SizedBox(width: 20.w), // Add spacing between text  
// Column(  
// crossAxisAlignment: CrossAxisAlignment.start,  
// children: [  
// Text(  
// 'Successfully Uploaded \nThe Claim Details ',  
// style: TextStyle(  
// color: Colors.black,  
// fontSize: 20.sp,  
// fontWeight: FontWeight.bold,  
// ),  
// ),  
// Text(  
// 'Thank You',  
// style: TextStyle(  
// color: Colors.black,  
// fontSize: 20.sp,  
// fontWeight: FontWeight.bold,  
// ),  
// ),  
// Text(  
// 'Press the Button to go Home',  
// style: TextStyle(  
// color: Colors.black,  
// fontSize: 20.sp,  
// fontWeight: FontWeight.bold,  
// ),  
// ),  
// ],  
// ),  
// ],  
// ),  
// ElevatedButton(  
// onPressed: () {  
// Navigator.of(context).pop();  
// Navigator.of(context).pop();  
// },  
// style: ElevatedButton.styleFrom(  
// backgroundColor: Colors.grey, // Set button color to grey  
// shape: RoundedRectangleBorder(  
// borderRadius: BorderRadius.circular(30.r),  
// ),  
// ),  
// child: SizedBox(  
// width: 200.w,  
// height: 70.h,  
// child: Center(  
// child: Text(  
// "Go Home",  
// style: TextStyle(color: Colors.white),  
// ),  
// ),  
// ),  
// ),  
// ],  
// ),  
// ),  
//  
// ),  
// );  
// }  
// }  
import 'package:cloud\_firestore/cloud\_firestore.dart';  
import 'package:flutter/material.dart';  
import 'package:geocoding/geocoding.dart';  
  
class DeviceDetailsScreen extends StatefulWidget {  
 final Stream<QuerySnapshot> devicedetails;  
 final String deviceId;  
  
 const DeviceDetailsScreen({required this.devicedetails, required this.deviceId, super.key});  
  
 @override  
 \_DeviceDetailsScreenState createState() => \_DeviceDetailsScreenState();  
}  
  
class \_DeviceDetailsScreenState extends State<DeviceDetailsScreen> {  
 String placeName = "Fetching location...";  
  
 @override  
 void initState() {  
 super.initState();  
 fetchPlaceName(); // Call the function when the widget initializes  
 }  
  
 void fetchPlaceName() async {  
 widget.devicedetails.listen((snapshot) {  
 for (var deviceData in snapshot.docs) {  
 if (widget.deviceId == (deviceData['deviceid'] ?? '')) {  
 var locationData = deviceData.data() as Map<String, dynamic>? ?? {};  
 if (locationData.containsKey('location')) {  
 var loc = locationData['location'];  
 if (loc is Map<String, dynamic>) {  
 double? latitude = (loc['latitude'] as num?)?.toDouble();  
 double? longitude = (loc['longitude'] as num?)?.toDouble();  
  
 if (latitude != null && longitude != null) {  
 placemarkFromCoordinates(latitude, longitude).then((placemarks) {  
 if (placemarks.isNotEmpty) {  
 Placemark place = placemarks.first;  
 String formattedPlace = "${place.locality ?? ''}, ${place.administrativeArea ?? ''}, ${place.country ?? ''}".trim();  
 setState(() {  
 placeName = formattedPlace.isNotEmpty ? formattedPlace : "Unknown Location";  
 });  
 }  
 }).catchError((e) {  
 setState(() {  
 placeName = "Location not found";  
 });  
 });  
 }  
 }  
 }  
 }  
 }  
 });  
 }  
  
 @override  
 Widget build(BuildContext context) {  
 return StreamBuilder<QuerySnapshot>(  
 stream: widget.devicedetails,  
 builder: (context, snapshot) {  
 if (snapshot.connectionState == ConnectionState.waiting) {  
 return Center(child: CircularProgressIndicator());  
 }  
  
 if (snapshot.hasError) {  
 return Center(child: Text('Error: ${snapshot.error}'));  
 }  
  
 if (!snapshot.hasData || snapshot.data!.docs.isEmpty) {  
 return Center(child: Text('No data available'));  
 }  
  
 for (var deviceData in snapshot.data!.docs) {  
 if (widget.deviceId == (deviceData['deviceid'] ?? '')) {  
 return Column(  
 crossAxisAlignment: CrossAxisAlignment.start,  
 children: [  
 SizedBox(height: 20),  
 Center(  
 child: Text(  
 "Device Details",  
 style: TextStyle(color: Colors.*black*, fontSize: 22, fontWeight: FontWeight.*bold*),  
 ),  
 ),  
 Text("EngineNo : ${deviceData['EngineNo'] ?? 'N/A'}",  
 style: TextStyle(color: Colors.*black*, fontSize: 20, fontWeight: FontWeight.*bold*)),  
 Text("Chassis Number : ${deviceData['chassis\_no'] ?? 'N/A'}",  
 style: TextStyle(color: Colors.*black*, fontSize: 20, fontWeight: FontWeight.*bold*)),  
 Text("Speed : ${deviceData['Speed'] ?? 'N/A'}",  
 style: TextStyle(color: Colors.*black*, fontSize: 20, fontWeight: FontWeight.*bold*)),  
 Text("Accident Location: $placeName",  
 style: TextStyle(color: Colors.*black*, fontSize: 20, fontWeight: FontWeight.*bold*)),  
 Text("Engine Temp : ${deviceData['EngineTemp'] ?? 'N/A'}",  
 style: TextStyle(color: Colors.*black*, fontSize: 20, fontWeight: FontWeight.*bold*)),  
 Text("Date : ${deviceData['date'] ?? 'N/A'}",  
 style: TextStyle(color: Colors.*black*, fontSize: 20, fontWeight: FontWeight.*bold*)),  
 Text("Accident Time : ${deviceData['time'] ?? 'N/A'}",  
 style: TextStyle(color: Colors.*black*, fontSize: 20, fontWeight: FontWeight.*bold*)),  
 ],  
 );  
 }  
 }  
  
 return SizedBox(); // No matching device found  
 },  
 );  
 }  
}