import 'package:cloud\_firestore/cloud\_firestore.dart';  
import 'package:flutter/material.dart';  
import 'package:flutter/services.dart';  
import 'package:flutter\_screenutil/flutter\_screenutil.dart';  
import 'package:pdf/pdf.dart';  
import 'package:pdf/widgets.dart' as pw;  
import 'package:path\_provider/path\_provider.dart';  
import 'dart:io';  
import 'package:permission\_handler/permission\_handler.dart';  
import 'package:flutter/foundation.dart'; // Import this to check platforms  
import 'dart:html' as html;  
  
class Status extends StatefulWidget {  
 final String id;  
  
 const Status({super.key, required this.id});  
  
 @override  
 State<Status> createState() => \_StatusState();  
}  
  
class \_StatusState extends State<Status> {  
 final firestore =  
 FirebaseFirestore.*instance*.collection("Requests").snapshots();  
  
 // Dummy data for the labels (to match with Firestore fields)  
 List<String> data = [  
 'Name',  
 'Email',  
 'Address',  
 'PhoneNumber',  
 'RegId',  
 'DeviceId',  
 'InsuranceType',  
 'VehicleNumber',  
 'TypeOfAdmission',  
 'HospitalNumber',  
 'HospitalName',  
 'HospitalIpNumber',  
 'FirNumber',  
 'Gender',  
 'BillNumber',  
 'ApprovalStatus',  
 'Amount',  
 'Reason'  
 ];  
  
 Future<void> requestStoragePermission() async {  
 if (!kIsWeb) {  
 // Only request permission on mobile platforms (not Web)  
 PermissionStatus status = await Permission.*storage*.request();  
  
 if (status.isGranted) {  
 print('Permission granted');  
 } else {  
 print('Permission denied');  
 }  
 }  
 }  
  
 Future<void> generatePdf(Map<String, dynamic> data) async {  
 await requestStoragePermission();  
  
 final pdf = pw.Document();  
  
 // Load the image from assets  
 final imageProvider = await \_loadImageFromAsset('assets/damaged\_car.jpg');  
  
 // Add the first page  
 pdf.addPage(pw.Page(  
 pageFormat: PdfPageFormat.*a4*,  
 build: (pw.Context context) {  
 return pw.Column(  
 crossAxisAlignment: pw.CrossAxisAlignment.start,  
 children: [  
 // Title and initial content  
 pw.Center(  
 child: pw.Text('Claim Insurance Application',  
 style: pw.TextStyle(fontSize: 28.sp)),  
 ),  
 pw.SizedBox(height: 5.h),  
 pw.Center(  
 child: pw.Text('TO BE FILLED IN BY THE HOSPITAL',  
 style: pw.TextStyle(fontSize: 15.sp)),  
 ),  
 pw.SizedBox(height: 5.h),  
 pw.Center(  
 child: pw.Text(  
 'The issue of the form is not be taken as an admission of liability',  
 style: pw.TextStyle(fontSize: 15.sp)),  
 ),  
 pw.SizedBox(height: 5.h),  
 pw.Center(  
 child: pw.Text(  
 'Please include the original preauthorization request form in lieu of PART A',  
 style: pw.TextStyle(fontSize: 12.sp)),  
 ),  
 pw.SizedBox(height: 5.h),  
 pw.Divider(),  
 pw.SizedBox(height: 10.h),  
  
 // Details section  
 pw.Text('Details',  
 style: pw.TextStyle(  
 color: PdfColors.*black*, fontSize: 24.sp)),  
 pw.SizedBox(height: 16.h),  
 pw.Text('Name: ${data['Name'] ?? 'N/A'}',  
 style: pw.TextStyle(fontSize: 18.sp)),  
 pw.SizedBox(height: 10.h),  
 pw.Text('Email: ${data['Email'] ?? 'N/A'}',  
 style: pw.TextStyle(fontSize: 18.sp)),  
 pw.SizedBox(height: 10.h),  
 pw.Text('Address: ${data['address'] ?? 'N/A'}',  
 style: pw.TextStyle(fontSize: 18.sp)),  
 pw.SizedBox(height: 10.h),  
 pw.Text('Phone Number: ${data['Phone'] ?? 'N/A'}',  
 style: pw.TextStyle(fontSize: 18.sp)),  
 pw.SizedBox(height: 10.h),  
 pw.Text('Gender: ${data['gender'] ?? 'N/A'}',  
 style: pw.TextStyle(fontSize: 18.sp)),  
 pw.SizedBox(height: 10.h),  
 pw.Text('Registration Id: ${data['regId'] ?? 'N/A'}',  
 style: pw.TextStyle(fontSize: 18.sp)),  
 pw.SizedBox(height: 10.h),  
  
 // Additional data section (insurance, hospital, etc.)  
 pw.Text('Insurance Type: ${data['InsuranceType'] ?? 'N/A'}',  
 style: pw.TextStyle(fontSize: 18.sp)),  
 pw.SizedBox(height: 10.h),  
 pw.Text('Device Id: ${data['deviceId'] ?? 'N/A'}',  
 style: pw.TextStyle(fontSize: 18.sp)),  
 pw.SizedBox(height: 10.h),  
 pw.Text('Hospital Name: ${data['hospitalName'] ?? 'N/A'}',  
 style: pw.TextStyle(fontSize: 18.sp)),  
 pw.SizedBox(height: 10.h),  
 pw.Text('Hospital IP Number: ${data['hospitalIpNumber'] ?? 'N/A'}',  
 style: pw.TextStyle(fontSize: 18.sp)),  
 pw.SizedBox(height: 10.h),  
 pw.Text('Hospital Number: ${data['hospitalNumber'] ?? 'N/A'}',  
 style: pw.TextStyle(fontSize: 18.sp)),  
 pw.SizedBox(height: 10.h),  
 pw.Text('Type of Admission: ${data['typeOfAdmission'] ?? 'N/A'}',  
 style: pw.TextStyle(fontSize: 18.sp)),  
 pw.SizedBox(height: 10.h),  
 pw.Text('FIR Number: ${data['firNumber'] ?? 'N/A'}',  
 style: pw.TextStyle(fontSize: 18.sp)),  
 pw.SizedBox(height: 10.h),  
 pw.Text('Bill Number: ${data['billNumber'] ?? 'N/A'}',  
 style: pw.TextStyle(fontSize: 18.sp)),  
 pw.SizedBox(height: 10.h),  
 pw.Text('Amount: ${data['amount'] ?? 'N/A'}',  
 style: pw.TextStyle(fontSize: 18.sp)),  
 pw.SizedBox(height: 10.h),  
 pw.Text('Approval Status: ${data['aprovelstatus'] ?? 'N/A'}',  
 style: pw.TextStyle(fontSize: 18.sp)),  
 pw.SizedBox(height: 10.h),  
 data['reason'] == null || data['reason'].trim().isEmpty  
 ? pw.SizedBox()  
 : pw.Text('Reason: ${data['reason'] ?? 'N/A'}',  
 style: pw.TextStyle(fontSize: 18.sp)),  
 ],  
 );  
 },  
 ));  
  
 // Add the second page for Device details  
 pdf.addPage(pw.Page(  
 pageFormat: PdfPageFormat.*a4*,  
 build: (pw.Context context) {  
 return pw.Column(  
 crossAxisAlignment: pw.CrossAxisAlignment.start,  
 children: [  
 pw.Text('Device Details',  
 style: pw.TextStyle(color: PdfColors.*black*, fontSize: 24.sp)),  
 pw.SizedBox(height: 16.h),  
 pw.Divider(),  
 pw.SizedBox(height: 10.h),  
 pw.Text('Engine No: ${data['engineNumber'] ?? 'N/A'}',  
 style: pw.TextStyle(fontSize: 18.sp)),  
 pw.SizedBox(height: 10.h),  
 pw.Text('Cheers No: ${data['cheersNumber'] ?? 'N/A'}',  
 style: pw.TextStyle(fontSize: 18.sp)),  
 pw.SizedBox(height: 10.h),  
 pw.Text('Temperature: ${data['engineTemperature'] ?? 'N/A'}',  
 style: pw.TextStyle(fontSize: 18.sp)),  
 pw.SizedBox(height: 10.h),  
 pw.Text('Speed: ${data['speed'] ?? 'N/A'}',  
 style: pw.TextStyle(fontSize: 18.sp)),  
 pw.SizedBox(height: 10.h),  
 pw.Text('Location: ${data['accidentLocation'] ?? 'N/A'}',  
 style: pw.TextStyle(fontSize: 18.sp)),  
 pw.SizedBox(height: 10.h),  
 pw.Text('Date: ${data['date'] ?? 'N/A'}',  
 style: pw.TextStyle(fontSize: 18.sp)), pw.SizedBox(height: 10.h),  
 pw.Text('Accident Time: ${data['accidentTime'] ?? 'N/A'}',  
 style: pw.TextStyle(fontSize: 18.sp)),  
 pw.SizedBox(height: 20.h),  
  
 pw.Image(imageProvider,  
 width: 200.w, height: 250.h, fit: pw.BoxFit.fill),  
 ],  
 );  
 },  
 ));  
  
 final pdfBytes = await pdf.save();  
  
 if (pdfBytes.isEmpty) {  
 print('Generated PDF is empty!');  
 return;  
 }  
  
 if (!kIsWeb) {  
 final outputDir = await getApplicationDocumentsDirectory();  
 final file = File("${outputDir.path}/claim\_application.pdf");  
  
 await file.writeAsBytes(pdfBytes);  
  
 if (await file.exists()) {  
 print("PDF saved successfully at: ${file.path}");  
 } else {  
 print("Failed to save PDF.");  
 }  
 } else {  
 final blob = html.Blob([pdfBytes]);  
 final url = html.Url.*createObjectUrlFromBlob*(blob);  
 final anchor = html.AnchorElement(href: url)  
 ..target = 'blank'  
 ..download = 'claim\_application.pdf'  
 ..click();  
 html.Url.*revokeObjectUrl*(url);  
 }  
 }  
  
 Future<pw.ImageProvider> \_loadImageFromAsset(String assetPath) async {  
 final byteData = await rootBundle.load(assetPath);  
 final imageBytes = byteData.buffer.asUint8List();  
 return pw.MemoryImage(imageBytes);  
 }  
  
 @override  
 Widget build(BuildContext context) {  
 return Scaffold(  
 appBar: AppBar(  
 backgroundColor: Colors.*white*,  
 title: Text(  
 'Claim Status',  
 style: TextStyle(color: Colors.*black*),  
 ),  
 elevation: 0,  
 ),  
 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: Padding(  
 padding: EdgeInsets.only(left: 100.w, right: 100.w),  
 child: StreamBuilder<QuerySnapshot>(  
 stream: firestore,  
 builder: (BuildContext context,  
 AsyncSnapshot<QuerySnapshot> snapshot) {  
 if (!snapshot.hasData) {  
 return Center(child: CircularProgressIndicator());  
 }  
 if (snapshot.hasError) {  
 return Center(  
 child: Text(  
 "ERROR",  
 style: TextStyle(color: Colors.*red*),  
 ),  
 );  
 }  
 if (snapshot.hasData) {  
 var matchingDoc = snapshot.data!.docs.where((doc) => doc.id == widget.id).toList();  
  
 if (matchingDoc.isEmpty) {  
 return SizedBox(); // or return a message that no matching document was found  
 }  
  
 // Use the first (and hopefully only) matching document  
 var document = matchingDoc.first;  
  
  
 return  
  
 // document['id'].toString()!=widget.id? SizedBox():  
  
  
 Container(  
 width: 600.w,  
 height: 800.h,  
 color: Colors.*grey*.withOpacity(0.4),  
 child: Padding(  
 padding: EdgeInsets.only(left: 20.w, right: 20.w),  
 child: SingleChildScrollView(  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.start,  
 children: [  
 SizedBox(height: 20.h),  
 Center(  
 child: Text(  
 "Claim Insurance Application",  
 style: TextStyle(  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 fontSize: 30.sp),  
 ),  
 ),  
 SizedBox(height: 20.h),  
 Divider(  
 thickness: 3.w,  
 color: Colors.*black*,  
 ),  
 SizedBox(height: 20.h),  
 Text(  
 'Claim Details:',  
 style: TextStyle(  
 fontSize: 22.sp,  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 ),  
 ),  
 SizedBox(height: 20.h),  
  
 Column(  
 crossAxisAlignment: CrossAxisAlignment.start,  
 children: [  
 Row(  
 children: [  
 SizedBox(  
 width: 160.w,  
 height: 30.h,  
 child: Text(  
 "Name :",  
 style: TextStyle(  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 fontSize: 18.sp,  
 ),  
 ),  
 ),  
 Text(  
 "${document['Name'] ?? 'N/A'}",  
 style: TextStyle(  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 fontSize: 18.sp,  
 ),  
 ),  
 ],  
 ),  
 SizedBox(height: 10.h),  
 Row(  
 children: [  
 SizedBox(  
 width: 160.w,  
 height: 30.h,  
 child: Text(  
 "Email :",  
 style: TextStyle(  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 fontSize: 18.sp,  
 ),  
 ),  
 ),  
 Text(  
 " ${document['Email'] ?? 'N/A'}",  
 style: TextStyle(  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 fontSize: 18.sp,  
 ),  
 ),  
 ],  
 ),  
 SizedBox(height: 10.h),  
 Row(  
 children: [  
 SizedBox(  
 width: 160.w,  
 height: 30.h,  
 child: Text(  
 "Address :",  
 style: TextStyle(  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 fontSize: 18.sp,  
 ),  
 ),  
 ),  
 Text(  
 " ${document['address'] ?? 'N/A'}",  
 style: TextStyle(  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 fontSize: 20.sp,  
 ),  
 ),  
 ],  
 ),  
 SizedBox(height: 10.h),  
 Row(  
 children: [  
 SizedBox(  
 width: 160.w,  
 height: 30.h,  
 child: Text(  
 "Phone :",  
 style: TextStyle(  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 fontSize: 18.sp,  
 ),  
 ),  
 ),  
 Text(  
 "${document['Phone'] ?? 'N/A'}",  
 style: TextStyle(  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 fontSize: 20.sp,  
 ),  
 ),  
 ],  
 ),  
 SizedBox(height: 10.h),  
 Row(  
 children: [  
 SizedBox(  
 width: 160.w,  
 height: 30.h,  
 child: Text(  
 "Reg Id :",  
 style: TextStyle(  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 fontSize: 18.sp,  
 ),  
 ),  
 ),  
 SizedBox(  
 width: 550.w,  
 child: Text(  
 " ${document['regId'] ?? 'N/A'}",  
 style: TextStyle(  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 fontSize: 20.sp,  
 ),  
 ),  
 ),  
 ],  
 ),  
 SizedBox(height: 10.h),  
 Row(  
 children: [  
 SizedBox(  
 width: 160.w,  
 height: 30.h,  
 child: Text(  
 "Device Id :",  
 style: TextStyle(  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 fontSize: 18.sp,  
 ),  
 ),  
 ),  
 Text(  
 "${document['deviceId'] ?? 'N/A'}",  
 style: TextStyle(  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 fontSize: 20.sp,  
 ),  
 ),  
 ],  
 ),  
 SizedBox(height: 10.h),  
 Row(  
 children: [  
 SizedBox(  
 width: 160.w,  
 height: 30.h,  
 child: Text(  
 "Insurance Type :",  
 style: TextStyle(  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 fontSize: 18.sp,  
 ),  
 ),  
 ),  
 Text(  
 " ${document['InsuranceType'] ?? 'N/A'}",  
 style: TextStyle(  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 fontSize: 20.sp,  
 ),  
 ),  
 ],  
 ),  
 SizedBox(height: 10.h),  
 Row(  
 children: [  
 SizedBox(  
 width: 160.w,  
 height: 30.h,  
 child: Text(  
 "Vehicle Number:",  
 style: TextStyle(  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 fontSize: 18.sp,  
 ),  
 ),  
 ),  
 Text(  
 " ${document['vehicleNumber'] ?? 'N/A'}",  
 style: TextStyle(  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 fontSize: 20.sp,  
 ),  
 ),  
 ],  
 ),  
 SizedBox(height: 20.h),  
 Row(  
 children: [  
 SizedBox(  
 width: 160.w,  
 height: 30.h,  
 child: Text(  
 "Admission Type :",  
 style: TextStyle(  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 fontSize: 18.sp,  
 ),  
 ),  
 ),  
 Text(  
 " ${document['typeOfAdmission'] ?? 'N/A'}",  
 style: TextStyle(  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 fontSize: 20.sp,  
 ),  
 ),  
 ],  
 ),  
 SizedBox(height: 10.h),  
 Row(  
 children: [  
 SizedBox(  
 width: 160.w,  
 height: 30.h,  
 child: Text(  
 "Hospital No :",  
 style: TextStyle(  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 fontSize: 18.sp,  
 ),  
 ),  
 ),  
 Text(  
 " ${document['hospitalNumber'] ?? 'N/A'}",  
 style: TextStyle(  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 fontSize: 20.sp,  
 ),  
 ),  
 ],  
 ),  
 SizedBox(height: 10.h),  
 Row(  
 children: [  
 SizedBox(  
 width: 160.w,  
 height: 30.h,  
 child: Text(  
 "Hospital Name :",  
 style: TextStyle(  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 fontSize: 18.sp,  
 ),  
 ),  
 ),  
 Text(  
 " ${document['hospitalName'] ?? 'N/A'}",  
 style: TextStyle(  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 fontSize: 20.sp,  
 ),  
 ),  
 ],  
 ),  
 SizedBox(height: 10.h),  
 Row(  
 children: [  
 SizedBox(  
 width: 160.w,  
 height: 30.h,  
 child: Text(  
 "Hospital Ip :",  
 style: TextStyle(  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 fontSize: 18.sp,  
 ),  
 ),  
 ),  
 Text(  
 " ${document['hospitalIpNumber'] ?? 'N/A'}",  
 style: TextStyle(  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 fontSize: 20.sp,  
 ),  
 ),  
 ],  
 ),  
 SizedBox(height: 10.h),  
 Row(  
 children: [  
 SizedBox(  
 width: 160.w,  
 height: 30.h,  
 child: Text(  
 "Fir No:",  
 style: TextStyle(  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 fontSize: 18.sp,  
 ),  
 ),  
 ),  
 Text(  
 " ${document['firNumber'] ?? 'N/A'}",  
 style: TextStyle(  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 fontSize: 20.sp,  
 ),  
 ),  
 ],  
 ),  
 SizedBox(height: 10.h),  
 Row(  
 children: [  
 SizedBox(  
 width: 160.w,  
 height: 30.h,  
 child: Text(  
 "Gender :",  
 style: TextStyle(  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 fontSize: 18.sp,  
 ),  
 ),  
 ),  
 Text(  
 " ${document['gender'] ?? 'N/A'}",  
 style: TextStyle(  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 fontSize: 20.sp,  
 ),  
 ),  
 ],  
 ),  
 SizedBox(height: 10.h),  
 Row(  
 children: [  
 SizedBox(  
 width: 160.w,  
 height: 30.h,  
 child: Text(  
 "Bill No :",  
 style: TextStyle(  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 fontSize: 18.sp,  
 ),  
 ),  
 ),  
 Text(  
 " ${document['billNumber'] ?? 'N/A'}",  
 style: TextStyle(  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 fontSize: 20.sp,  
 ),  
 ),  
 ],  
 ),  
 SizedBox(height: 10.h),  
 Row(  
 children: [  
 SizedBox(  
 width: 160.w,  
 height: 30.h,  
 child: Text(  
 "Status :",  
 style: TextStyle(  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 fontSize: 18.sp,  
 ),  
 ),  
 ),  
 Text(  
 " ${document['aprovelstatus'] ?? 'N/A'}",  
 style: TextStyle(  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 fontSize: 20.sp,  
 ),  
 ),  
 ],  
 ),  
 SizedBox(height: 10.h),  
 Row(  
 children: [  
 SizedBox(  
 width: 160.w,  
 height: 30.h,  
 child: Text(  
 "Amount :",  
 style: TextStyle(  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 fontSize: 18.sp,  
 ),  
 ),  
 ),  
 Text(  
 " ${document['amount'] ?? 'N/A'}",  
 style: TextStyle(  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 fontSize: 20.sp,  
 ),  
 ),  
 ],  
 ),  
 SizedBox(height: 10.h),  
  
 document['reason'].toString().isEmpty||document['reason']=='' || document['reason']==null? SizedBox():  
 Row(  
 children: [  
 SizedBox(  
 width: 160.w,  
 height: 30.h,  
 child: Text(  
 "Reason :",  
 style: TextStyle(  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 fontSize: 18.sp,  
 ),  
 ),  
 ),  
 Text(  
 " ${document['reason'] ?? 'N/A'}",  
 style: TextStyle(  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 fontSize: 20.sp,  
 ),  
 ),  
 ],  
 ),  
 SizedBox(height: 20.h),  
 Center(  
 child: Padding(  
 padding: EdgeInsets.only(left: 20.w),  
 child: Text(  
 'Device Details',  
 style: TextStyle(  
 fontSize: 22.sp,  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 ),  
 ),  
 ),  
 ),  
 SizedBox(height: 20.h),  
 Padding(  
 padding: EdgeInsets.only(right: 20.w),  
 child: Divider(  
 thickness: 3.w,  
 color: Colors.*black*,  
 ),  
 ),  
 SizedBox(height: 20.h),  
 Row(crossAxisAlignment: CrossAxisAlignment.start,  
 children: [  
 Column(crossAxisAlignment: CrossAxisAlignment.start,  
 children: [  
  
 Row(  
 children: [  
 SizedBox(  
 width: 160.w,  
 height: 30.h,  
 child: Text(  
 "Engine No :",  
 style: TextStyle(  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 fontSize: 18.sp,  
 ),  
 ),  
 ),  
 Text(  
 " ${document['engineNumber'] ?? 'N/A'}",  
 style: TextStyle(  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 fontSize: 20.sp,  
 ),  
 ),  
 ],  
 ),  
 SizedBox(height: 10.h),  
 Row(  
 children: [  
 SizedBox(  
 width: 160.w,  
 height: 30.h,  
 child: Text(  
 "Cheers No :",  
 style: TextStyle(  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 fontSize: 18.sp,  
 ),  
 ),  
 ),  
 Text(  
 " ${document['chassisNumber'] ?? 'N/A'}",  
 style: TextStyle(  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 fontSize: 20.sp,  
 ),  
 ),  
 ],  
 ),  
 SizedBox(height: 10.h),  
 Row(  
 children: [  
 SizedBox(  
 width: 160.w,  
 height: 30.h,  
 child: Text(  
 "Temperature :",  
 style: TextStyle(  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 fontSize: 18.sp,  
 ),  
 ),  
 ),  
 Text(  
 " ${document['engineTemperature'] ?? 'N/A'}",  
 style: TextStyle(  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 fontSize: 20.sp,  
 ),  
 ),  
 ],  
 ),  
 SizedBox(height: 10.h),  
 Row(  
 children: [  
 SizedBox(  
 width: 160.w,  
 height: 30.h,  
 child: Text(  
 "Speed :",  
 style: TextStyle(  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 fontSize: 18.sp,  
 ),  
 ),  
 ),  
 Text(  
 " ${document['speed'] ?? 'N/A'}",  
 style: TextStyle(  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 fontSize: 20.sp,  
 ),  
 ),  
 ],  
 ),  
 SizedBox(height: 10.h),  
 Row(  
 children: [  
 SizedBox(  
 width: 160.w,  
 height: 30.h,  
 child: Text(  
 " Location :",  
 style: TextStyle(  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 fontSize: 18.sp,  
 ),  
 ),  
 ),  
 Text(  
 " ${document['accidentLocation'] ?? 'N/A'}",  
 style: TextStyle(  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 fontSize: 20.sp,  
 ),  
 ),  
 ],  
 ),  
 SizedBox(height: 10.h),  
 Row(  
 children: [  
 SizedBox(  
 width: 160.w,  
 height: 30.h,  
 child: Text(  
 "Date :",  
 style: TextStyle(  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 fontSize: 18.sp,  
 ),  
 ),  
 ),  
 Text(  
 " ${document['date'] ?? 'N/A'}",  
 style: TextStyle(  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 fontSize: 20.sp,  
 ),  
 ),  
 ],  
 ),  
 SizedBox(height: 10.h),  
 Row(  
 children: [  
 SizedBox(  
 width: 160.w,  
 height: 30.h,  
 child: Text(  
 "Accident Time :",  
 style: TextStyle(  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 fontSize: 18.sp,  
 ),  
 ),  
 ),  
 Text(  
 " ${document['accidentTime'] ?? 'N/A'}",  
 style: TextStyle(  
 color: Colors.*black*,  
 fontWeight: FontWeight.*bold*,  
 fontSize: 20.sp,  
 ),  
 ),  
 ],  
 ),  
 SizedBox(height: 10.h),  
 ],  
 ), SizedBox(width: 100.w,),  
 Image.asset(  
 "assets/damaged\_car.jpg",  
 width: 200.w,  
 height: 250.h,  
 fit: BoxFit.fill,  
 ),  
 ],  
 )  
 ],  
 ),  
  
  
  
 SizedBox(height: 30.h  
 ,),  
 Center(  
 child: GestureDetector(  
 onTap: () {  
 FirebaseFirestore.*instance* .collection("Requests")  
 .doc(widget.id)  
 .get()  
 .then((document) {  
 if (document.exists) {  
 generatePdf(document.data()!);  
 }  
 });  
 },  
 child: Container(  
 width: 200.w,  
 height: 60.h,  
 decoration: ShapeDecoration(  
 color: Colors.*white*,  
 shape: RoundedRectangleBorder(  
 borderRadius: BorderRadius.circular(10.r),  
 ),  
 ),  
 child: Center(  
 child: Text(  
 "Print",  
 style: TextStyle(  
 color: Colors.*black*, fontSize: 20.sp),  
 ),  
 ),  
 ),  
 ),  
 ),  
 SizedBox(height: 30.h),  
 ],  
 ),  
 ),  
 ),  
 );  
 }else{return SizedBox(); }}  
 ),  
 ),  
 ),  
 );  
 }  
}