CODE FOR PREP-MUN APPLICATION.

Profile.dart:

import 'package:flutter/material.dart';  
import 'package:firebase\_auth/firebase\_auth.dart';  
import 'package:intl/intl.dart';  
  
import 'authentication/authstream.dart';  
  
class ProfileScreen extends StatelessWidget {  
 final String userEmail;  
 final String userName;  
  
 const ProfileScreen({  
 Key? key,  
 required this.userEmail,  
 required this.userName,  
 }) : super(key: key);  
  
 @override  
 Widget build(BuildContext context) {  
 // Mock data for registered conferences. Replace with your database logic.  
 final List<Map<String, String>> registeredConferences = [  
 {'title': 'Flutter Dev Conference', 'date': '2025-01-20'},  
 {'title': 'AI Summit 2025', 'date': '2025-02-15'},  
 ];  
  
 return Scaffold(  
 backgroundColor: Colors.grey[100],  
 appBar: AppBar(  
 title: const Text('Profile'),  
 backgroundColor: Colors.black,  
 elevation: 0,  
 actions: [  
 IconButton(  
 icon: const Icon(Icons.logout),  
 onPressed: ()  
 {  
 FirebaseAuth \_auth = FirebaseAuth.instance;  
 \_auth.signOut();  
 Navigator.push(context, MaterialPageRoute(builder: (context) => Authstream()));  
 },  
 ),  
 ],  
 ),  
 body: SingleChildScrollView(  
 child: Column(  
 children: [  
 // Profile Header Section  
 Container(  
 decoration: BoxDecoration(  
 color: Colors.black,  
 borderRadius: const BorderRadius.only(  
 bottomLeft: Radius.circular(30),  
 bottomRight: Radius.circular(30),  
 ),  
 ),  
 child: Padding(  
 padding: const EdgeInsets.all(20.0),  
 child: Column(  
 children: [  
 const CircleAvatar(  
 radius: 50,  
 backgroundColor: Colors.white,  
 child: Icon(Icons.person, size: 50, color: Colors.black),  
 ),  
 const SizedBox(height: 16),  
 Text(  
 userName,  
 style: const TextStyle(  
 fontSize: 24,  
 fontWeight: FontWeight.bold,  
 color: Colors.white,  
 ),  
 ),  
 const SizedBox(height: 8),  
 Text(  
 userEmail,  
 style: const TextStyle(  
 fontSize: 16,  
 color: Colors.white70,  
 ),  
 ),  
 const SizedBox(height: 20),  
 ],  
 ),  
 ),  
 ),  
  
 // Conferences Section  
 Padding(  
 padding: const EdgeInsets.all(20.0),  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.start,  
 children: [  
 const Row(  
 mainAxisAlignment: MainAxisAlignment.spaceBetween,  
 children: [  
 Text(  
 'Registered Conferences',  
 style: TextStyle(  
 fontSize: 20,  
 fontWeight: FontWeight.bold,  
 color: Colors.black87,  
 ),  
 ),  
 Chip(  
 label: Text('2 Active'),  
 backgroundColor: Colors.black,  
 labelStyle: TextStyle(color: Colors.white),  
 ),  
 ],  
 ),  
 const SizedBox(height: 16),  
 ...registeredConferences.map((conference) {  
 final formattedDate = DateFormat('MMM dd, yyyy')  
 .format(DateTime.parse(conference['date']!));  
 return Card(  
 elevation: 2,  
 margin: const EdgeInsets.only(bottom: 16),  
 shape: RoundedRectangleBorder(  
 borderRadius: BorderRadius.circular(12),  
 ),  
 child: ListTile(  
 contentPadding: const EdgeInsets.all(16),  
 leading: Container(  
 width: 50,  
 height: 50,  
 decoration: BoxDecoration(  
 color: Colors.black.withOpacity(0.1),  
 borderRadius: BorderRadius.circular(8),  
 ),  
 child: const Icon(  
 Icons.event,  
 color: Colors.black,  
 ),  
 ),  
 title: Text(  
 conference['title']!,  
 style: const TextStyle(  
 fontSize: 18,  
 fontWeight: FontWeight.bold,  
 ),  
 ),  
 subtitle: Column(  
 crossAxisAlignment: CrossAxisAlignment.start,  
 children: [  
 const SizedBox(height: 8),  
 Row(  
 children: [  
 const Icon(Icons.calendar\_today,  
 size: 16,  
 color: Colors.grey),  
 const SizedBox(width: 8),  
 Text(  
 formattedDate,  
 style: const TextStyle(  
 color: Colors.grey,  
 ),  
 ),  
 ],  
 ),  
 ],  
 ),  
 trailing: const Icon(Icons.chevron\_right),  
 onTap: () {  
 // Add navigation to conference details  
 },  
 ),  
 );  
 }).toList(),  
 ],  
 ),  
 ),  
 ],  
 ),  
 ),  
 );  
 }  
}

Splash\_screen.dart:

import 'package:flutter/material.dart';  
import 'package:flutter/scheduler.dart';  
import 'package:mohsin/authentication/authstream.dart';  
import 'chat\_screen.dart';  
  
class SplashScreen extends StatefulWidget {  
 @override  
 \_SplashScreenState createState() => \_SplashScreenState();  
}  
  
class \_SplashScreenState extends State<SplashScreen>  
 with SingleTickerProviderStateMixin {  
 late AnimationController \_controller;  
  
 @override  
 void initState() {  
 super.initState();  
  
 // Initialize animation controller  
 \_controller = AnimationController(  
 vsync: this,  
 duration: const Duration(milliseconds: 900),  
 )..repeat(reverse: true);  
  
 SchedulerBinding.instance.addPostFrameCallback((\_) {  
 // Navigate after splash screen  
 Future.delayed(Duration(seconds: 3), () {  
 Navigator.pushReplacement(  
 context,  
 MaterialPageRoute(builder: (context) => Authstream()),  
 );  
 });  
 });  
 }  
  
 @override  
 void dispose() {  
 \_controller.dispose();  
 super.dispose();  
 }  
  
 @override  
 Widget build(BuildContext context) {  
 return Scaffold(  
 backgroundColor: Colors.grey,  
 body: Center(  
 child: Column(  
 mainAxisAlignment: MainAxisAlignment.center,  
 children: [  
 // Circular Avatar for Logo  
 CircleAvatar(  
 radius: 60,  
 backgroundColor: Colors.white,  
 child: ClipOval(  
 child: Image.asset(  
 'images/logo.jpg', // Replace with your logo path  
 fit: BoxFit.cover,  
 height: 100,  
 width: 100,  
 ),  
 ),  
 ),  
 const SizedBox(height: 15),  
  
 // "ChatBot" Text  
 const Text(  
 'MyMun',  
 style: TextStyle(  
 fontSize: 24,  
 fontWeight: FontWeight.bold,  
 color: Colors.white,  
 ),  
 ),  
 const SizedBox(height: 10),  
  
 // Vertical Dots Animation  
 AnimatedBuilder(  
 animation: \_controller,  
 builder: (context, child) {  
 return Row(  
 mainAxisAlignment: MainAxisAlignment.center,  
 children: List.generate(3, (index) {  
 return AnimatedOpacity(  
 opacity: \_controller.value > (index \* 0.3) ? 1.0 : 0.3,  
 duration: const Duration(milliseconds: 300),  
 child: const Padding(  
 padding: EdgeInsets.symmetric(horizontal: 4.0),  
 child: CircleAvatar(  
 radius: 5,  
 backgroundColor: Colors.white,  
 ),  
 ),  
 );  
 }),  
 );  
 },  
 ),  
 ],  
 ),  
 ),  
 );  
 }  
}

Pubspec:

*#name: mohsin  
#description: "A new Flutter project."  
## The following line prevents the package from being accidentally published to  
## pub.dev using `flutter pub publish`. This is preferred for private packages.  
#publish\_to: 'none' # Remove this line if you wish to publish to pub.dev  
#  
## The following defines the version and build number for your application.  
## A version number is three numbers separated by dots, like 1.2.43  
## followed by an optional build number separated by a +.  
## Both the version and the builder number may be overridden in flutter  
## build by specifying --build-name and --build-number, respectively.  
## In Android, build-name is used as versionName while build-number used as versionCode.  
## Read more about Android versioning at https://developer.android.com/studio/publish/versioning  
## In iOS, build-name is used as CFBundleShortVersionString while build-number is used as CFBundleVersion.  
## Read more about iOS versioning at  
## https://developer.apple.com/library/archive/documentation/General/Reference/InfoPlistKeyReference/Articles/CoreFoundationKeys.html  
## In Windows, build-name is used as the major, minor, and patch parts  
## of the product and file versions while build-number is used as the build suffix.  
#version: 1.0.0+1  
#  
#environment:  
# sdk: ^3.5.3  
#  
## Dependencies specify other packages that your package needs in order to work.  
## To automatically upgrade your package dependencies to the latest versions  
## consider running `flutter pub upgrade --major-versions`. Alternatively,  
## dependencies can be manually updated by changing the version numbers below to  
## the latest version available on pub.dev. To see which dependencies have newer  
## versions available, run `flutter pub outdated`.  
#dependencies:  
# flutter:  
# sdk: flutter  
#  
#  
# # The following adds the Cupertino Icons font to your application.  
# # Use with the CupertinoIcons class for iOS style icons.  
# cupertino\_icons: ^1.0.8  
# http: ^1.2.2  
# firebase\_auth: ^5.4.0  
# firebase\_core: ^3.10.0  
# cloud\_firestore: ^5.6.1  
# crypto: ^3.0.6  
# intl: ^0.20.1  
# flutter\_launcher\_icons: ^0.14.2  
# font\_awesome\_flutter: ^10.8.0  
#  
#dev\_dependencies:  
# flutter\_test:  
# sdk: flutter  
#  
# # The "flutter\_lints" package below contains a set of recommended lints to  
# # encourage good coding practices. The lint set provided by the package is  
# # activated in the `analysis\_options.yaml` file located at the root of your  
# # package. See that file for information about deactivating specific lint  
# # rules and activating additional ones.  
# flutter\_lints: ^4.0.0  
#  
## For information on the generic Dart part of this file, see the  
## following page: https://dart.dev/tools/pub/pubspec  
#  
## The following section is specific to Flutter packages.  
#flutter:  
#  
# # The following line ensures that the Material Icons font is  
# # included with your application, so that you can use the icons in  
# # the material Icons class.  
# uses-material-design: true  
#  
# # To add assets to your application, add an assets section, like this:  
# assets:  
# - assets/images/  
# # - imag es/a\_dot\_ham.jpeg  
#  
# # An image asset can refer to one or more resolution-specific "variants", see  
# # https://flutter.dev/to/resolution-aware-images  
#  
# # For details regarding adding assets from package dependencies, see  
# # https://flutter.dev/to/asset-from-package  
#  
# # To add custom fonts to your application, add a fonts section here,  
# # in this "flutter" section. Each entry in this list should have a  
# # "family" key with the font family name, and a "fonts" key with a  
# # list giving the asset and other descriptors for the font. For  
# # example:  
# # fonts:  
# # - family: Schyler  
# # fonts:  
# # - asset: fonts/Schyler-Regular.ttf  
# # - asset: fonts/Schyler-Italic.ttf  
# # style: italic  
# # - family: Trajan Pro  
# # fonts:  
# # - asset: fonts/TrajanPro.ttf  
# # - asset: fonts/TrajanPro\_Bold.ttf  
# # weight: 700  
# #  
# # For details regarding fonts from package dependencies,  
# # see https://flutter.dev/to/font-from-package  
#flutter\_icons:  
# android: true  
# image\_path: "assets/images/mymun\_logo\_for\_launch.jpeg"*name: mohsin  
description: "A new Flutter project."  
*# The following line prevents the package from being accidentally published to  
# pub.dev using `flutter pub publish`. This is preferred for private packages.*publish\_to: 'none' *# Remove this line if you wish to publish to pub.dev  
  
# The following defines the version and build number for your application.  
# A version number is three numbers separated by dots, like 1.2.43  
# followed by an optional build number separated by a +.  
# Both the version and the builder number may be overridden in flutter  
# build by specifying --build-name and --build-number, respectively.  
# In Android, build-name is used as versionName while build-number used as versionCode.  
# Read more about Android versioning at https://developer.android.com/studio/publish/versioning  
# In iOS, build-name is used as CFBundleShortVersionString while build-number is used as CFBundleVersion.  
# Read more about iOS versioning at  
# https://developer.apple.com/library/archive/documentation/General/Reference/InfoPlistKeyReference/Articles/CoreFoundationKeys.html  
# In Windows, build-name is used as the major, minor, and patch parts  
# of the product and file versions while build-number is used as the build suffix.*version: 1.0.0+1  
  
environment:  
 sdk: ^3.5.3  
  
*# Dependencies specify other packages that your package needs in order to work.  
# To automatically upgrade your package dependencies to the latest versions  
# consider running `flutter pub upgrade --major-versions`. Alternatively,  
# dependencies can be manually updated by changing the version numbers below to  
# the latest version available on pub.dev. To see which dependencies have newer  
# versions available, run `flutter pub outdated`.*dependencies:  
 flutter:  
 sdk: flutter  
  
  
 *# The following adds the Cupertino Icons font to your application.  
 # Use with the CupertinoIcons class for iOS style icons.* cupertino\_icons: ^1.0.8  
 http: ^1.2.2  
 firebase\_auth: ^5.4.0  
 firebase\_core: ^3.10.0  
 cloud\_firestore: ^5.6.1  
 crypto: ^3.0.6  
 intl:  
 flutter\_launcher\_icons: ^0.14.2  
 chat\_gpt\_sdk: ^3.1.4  
 dash\_chat\_2: ^0.0.21  
 font\_awesome\_flutter: ^10.8.0  
  
dev\_dependencies:  
 flutter\_test:  
 sdk: flutter  
  
 *# The "flutter\_lints" package below contains a set of recommended lints to  
 # encourage good coding practices. The lint set provided by the package is  
 # activated in the `analysis\_options.yaml` file located at the root of your  
 # package. See that file for information about deactivating specific lint  
 # rules and activating additional ones.* flutter\_lints: ^4.0.0  
  
*# For information on the generic Dart part of this file, see the  
# following page: https://dart.dev/tools/pub/pubspec  
  
# The following section is specific to Flutter packages.*flutter:  
  
 *# The following line ensures that the Material Icons font is  
 # included with your application, so that you can use the icons in  
 # the material Icons class.* uses-material-design: true  
  
 *# To add assets to your application, add an assets section, like this:* assets:  
 - assets/images/  
 *# - images/a\_dot\_ham.jpeg  
  
 # An image asset can refer to one or more resolution-specific "variants", see  
 # https://flutter.dev/to/resolution-aware-images  
  
 # For details regarding adding assets from package dependencies, see  
 # https://flutter.dev/to/asset-from-package  
  
 # To add custom fonts to your application, add a fonts section here,  
 # in this "flutter" section. Each entry in this list should have a  
 # "family" key with the font family name, and a "fonts" key with a  
 # list giving the asset and other descriptors for the font. For  
 # example:  
 # fonts:  
 # - family: Schyler  
 # fonts:  
 # - asset: fonts/Schyler-Regular.ttf  
 # - asset: fonts/Schyler-Italic.ttf  
 # style: italic  
 # - family: Trajan Pro  
 # fonts:  
 # - asset: fonts/TrajanPro.ttf  
 # - asset: fonts/TrajanPro\_Bold.ttf  
 # weight: 700  
 #  
 # For details regarding fonts from package dependencies,  
 # see https://flutter.dev/to/font-from-package*flutter\_icons:  
 android: true  
 image\_path: "assets/images/mymun\_logo\_for\_launch.jpeg"

//SPLASH SCREEN DART

import 'package:flutter/material.dart';  
import 'package:flutter/scheduler.dart';  
import 'package:mohsin/authentication/authstream.dart';  
import 'chat\_screen.dart';  
  
class SplashScreen extends StatefulWidget {  
 @override  
 \_SplashScreenState createState() => \_SplashScreenState();  
}  
  
class \_SplashScreenState extends State<SplashScreen>  
 with SingleTickerProviderStateMixin {  
 late AnimationController \_controller;  
  
 @override  
 void initState() {  
 super.initState();  
  
 // Initialize animation controller  
 \_controller = AnimationController(  
 vsync: this,  
 duration: const Duration(milliseconds: 900),  
 )..repeat(reverse: true);  
  
 SchedulerBinding.*instance*.addPostFrameCallback((\_) {  
 // Navigate after splash screen  
 Future.delayed(Duration(seconds: 3), () {  
 Navigator.*pushReplacement*(  
 context,  
 MaterialPageRoute(builder: (context) => Authstream()),  
 );  
 });  
 });  
 }  
  
 @override  
 void dispose() {  
 \_controller.dispose();  
 super.dispose();  
 }  
  
 @override  
 Widget build(BuildContext context) {  
 return Scaffold(  
 backgroundColor: Colors.*grey*,  
 body: Center(  
 child: Column(  
 mainAxisAlignment: MainAxisAlignment.center,  
 children: [  
 // Circular Avatar for Logo  
 CircleAvatar(  
 radius: 60,  
 backgroundColor: Colors.*white*,  
 child: ClipOval(  
 child: Image.asset(  
 'images/logo.jpg', // Replace with your logo path  
 fit: BoxFit.cover,  
 height: 100,  
 width: 100,  
 ),  
 ),  
 ),  
 const SizedBox(height: 15),  
  
 // "ChatBot" Text  
 const Text(  
 'MyMun',  
 style: TextStyle(  
 fontSize: 24,  
 fontWeight: FontWeight.*bold*,  
 color: Colors.*white*,  
 ),  
 ),  
 const SizedBox(height: 10),  
  
 // Vertical Dots Animation  
 AnimatedBuilder(  
 animation: \_controller,  
 builder: (context, child) {  
 return Row(  
 mainAxisAlignment: MainAxisAlignment.center,  
 children: List.generate(3, (index) {  
 return AnimatedOpacity(  
 opacity: \_controller.value > (index \* 0.3) ? 1.0 : 0.3,  
 duration: const Duration(milliseconds: 300),  
 child: const Padding(  
 padding: EdgeInsets.symmetric(horizontal: 4.0),  
 child: CircleAvatar(  
 radius: 5,  
 backgroundColor: Colors.*white*,  
 ),  
 ),  
 );  
 }),  
 );  
 },  
 ),  
 ],  
 ),  
 ),  
 );  
 }  
}

//BUILT GULDER

plugins **{** id "com.android.application"  
 // START: FlutterFire Configuration  
 id 'com.google.gms.google-services'  
 // END: FlutterFire Configuration  
 id "kotlin-android"  
 // The Flutter Gradle Plugin must be applied after the Android and Kotlin Gradle plugins.  
 id "dev.flutter.flutter-gradle-plugin"  
**}**android **{** namespace = "com.example.mohsin"  
 compileSdk = flutter.compileSdkVersion  
 ndkVersion = flutter.ndkVersion  
  
 compileOptions **{** sourceCompatibility = JavaVersion.VERSION\_1\_8  
 targetCompatibility = JavaVersion.VERSION\_1\_8  
 **}** kotlinOptions **{** jvmTarget = JavaVersion.VERSION\_1\_8  
 **}** defaultConfig **{** // *TODO: Specify your own unique Application ID (https://developer.android.com/studio/build/application-id.html).* applicationId = "com.example.mohsin"  
 // You can update the following values to match your application needs.  
 // For more information, see: https://flutter.dev/to/review-gradle-config.  
 minSdk = 23  
 targetSdk = flutter.targetSdkVersion  
 versionCode = flutter.versionCode  
 versionName = flutter.versionName  
 **}** buildTypes **{** release **{** // *TODO: Add your own signing config for the release build.* // Signing with the debug keys for now, so `flutter run --release` works.  
 signingConfig = signingConfigs.debug  
 **}  
 }  
}**flutter **{** source = "../.."  
**}**

//API DART SERVER

import 'dart:convert';  
import 'package:http/http.dart' as http;  
  
class ApiService {  
 static const String *\_baseUrl* = 'https://e64f-34-16-161-120.ngrok-free.app/chat'; // Replace with your backend URL  
  
 static Future<String> *getBotResponse*(String userInput) async {  
 try {  
 final response = await http.post(  
 Uri.*parse*(*\_baseUrl*),  
 headers: {'Content-Type': 'application/json'},  
 body: jsonEncode({'message': userInput}),  
 );  
  
 print('Response status: ${response.statusCode}');  
 print('Response body: ${response.body}');  
  
 if (response.statusCode == 200) {  
 final data = jsonDecode(response.body);  
 return data['response'] ?? 'No response from bot';  
 } else {  
 return 'Error: ${response.reasonPhrase}';  
 }  
 } catch (e) {  
 print('Error: $e');  
 return 'Error: Unable to connect to server';  
 }  
 }  
  
}

// BLOG DART

import 'package:flutter/material.dart';  
import 'package:font\_awesome\_flutter/font\_awesome\_flutter.dart';  
class Blogs extends StatefulWidget {  
 const Blogs({super.key});  
  
 @override  
 State<Blogs> createState() => \_BlogsState();  
}  
  
class \_BlogsState extends State<Blogs> {  
 @override  
 Widget build(BuildContext context) {  
 return Scaffold(  
 appBar: AppBar(  
 title: Text('MyMun Blogs',style: TextStyle(color: Colors.*white*,fontWeight: FontWeight.*bold*,letterSpacing: 1),),  
 centerTitle: true,  
 backgroundColor: Colors.*black*,  
 ),  
 backgroundColor: Colors.*white*,  
 body: Container(  
 height: MediaQuery.*of*(context).size.height,  
 width: MediaQuery.*of*(context).size.width,  
 child: SingleChildScrollView(  
 child: Column(  
 children: [  
 SizedBox(height: 15,),  
 Row(  
 mainAxisAlignment: MainAxisAlignment.center,  
 children: [  
 Padding(  
 padding: const EdgeInsets.all(8.0),  
 child: Container(  
 height: 60,  
 width: 220,  
 child: ElevatedButton.icon(  
 onPressed: () {  
 },  
 icon: Icon(  
 Icons.*border\_all*,  
 color: Colors.*white*,  
 ),  
 label: Text('All',  
 style: TextStyle(  
 color: Colors.*white*,  
 fontSize: 16.0,  
 ),  
 ),  
 style: ElevatedButton.*styleFrom*(  
 backgroundColor: Colors.*orange*,  
 foregroundColor: Colors.*orange*,  
 side: BorderSide(  
 color: Colors.*orange*,  
 width: 2.0,  
 ),  
 shape: RoundedRectangleBorder(  
 borderRadius: BorderRadius.circular(30.0),  
 ),  
 padding: EdgeInsets.symmetric(vertical: 10.0, horizontal: 20.0),  
 ),  
 ),  
 ),  
 ),  
 Padding(  
 padding: const EdgeInsets.all(8.0),  
 child: Container(  
 height: 60,  
 width: 220,  
 child: ElevatedButton.icon(  
 onPressed: () {},  
 icon: Icon(  
 Icons.*school*,  
 color: Colors.*green*,  
 ),  
 label: Text(  
 'Learning MUN',  
 style: TextStyle(  
 color: Colors.*green*,  
 fontSize: 16.0,  
 ),  
 ),  
 style: ElevatedButton.*styleFrom*(  
 backgroundColor: Colors.*white*,  
 foregroundColor: Colors.*green*,  
 side: BorderSide(  
 color: Colors.*green*,  
 width: 2.0,  
 ),  
 shape: RoundedRectangleBorder(  
 borderRadius: BorderRadius.circular(30.0),  
 ),  
 padding: EdgeInsets.symmetric(vertical: 10.0, horizontal: 20.0),  
 ),  
 ),  
 ),  
 ),  
 ],  
 ),  
 SizedBox(height: 7,),  
 Row(  
 mainAxisAlignment: MainAxisAlignment.center,  
 children: [  
 Padding(  
 padding: const EdgeInsets.all(8.0),  
 child: Container(  
 height: 60,  
 width: 220,  
 child: ElevatedButton.icon(  
 onPressed: () {},  
 icon: Icon(  
 Icons.*stars*, // Horoscope icon  
 color: Colors.*blue*, // Icon color  
 ),  
 label: Text(  
 'MUN Horoscope',  
 style: TextStyle(  
 color: Colors.*blue*, // Text color  
 fontSize: 16.0,  
 fontWeight: FontWeight.*bold*,  
 ),  
 ),  
 style: ElevatedButton.*styleFrom*(  
 backgroundColor: Colors.*white*, // Background color  
 foregroundColor: Colors.*blue*, // Foreground color  
 side: BorderSide(  
 color: Colors.*blue*, // Border color  
 width: 2.0,  
 ),  
 shape: RoundedRectangleBorder(  
 borderRadius: BorderRadius.circular(30.0), // Rounded corners  
 ),  
 fixedSize: Size(220.0, 60.0), // Button size  
 ),  
 ),  
 ),  
 ),  
 Padding(  
 padding: const EdgeInsets.all(8.0),  
 child: Container(  
 height: 60,  
 width: 220,  
 child: ElevatedButton.icon(  
 onPressed: () {  
 // Add your action here  
 },  
 icon: Icon(  
 Icons.*people*, // Suitable icon  
 color: Colors.*indigo*, // Icon color  
 ),  
 label: Text(  
 'mymun', // Lowercase text  
 style: TextStyle(  
 color: Colors.*indigo*, // Text color  
 fontSize: 16.0,  
 fontWeight: FontWeight.*bold*,  
 ),  
 ),  
 style: ElevatedButton.*styleFrom*(  
 backgroundColor: Colors.*white*, // Background color  
 foregroundColor: Colors.*indigo*, // Foreground color  
 side: BorderSide(  
 color: Colors.*indigo*, // Border color  
 width: 2.0,  
 ),  
 shape: RoundedRectangleBorder(  
 borderRadius: BorderRadius.circular(30.0), // Rounded corners  
 ),  
 fixedSize: Size(220.0, 60.0), // Button size  
 ),  
 ),  
 ),  
 )  
 ],  
 ),  
 SizedBox(height: 7,),  
 Row(  
 mainAxisAlignment: MainAxisAlignment.center,  
 children: [  
 Padding(  
 padding: const EdgeInsets.all(8.0),  
 child: Container(  
 height: 60,  
 width: 220,  
 child: ElevatedButton.icon(  
 onPressed: () {},  
 icon: Icon(  
 Icons.*mic*, // Mic icon for "MUN Stars"  
 color: Colors.*pink*.shade800, // Icon color  
 ),  
 label: Text(  
 'MUN Stars',  
 style: TextStyle(  
 color: Colors.*pink*.shade800, // Text color  
 fontSize: 16.0,  
 fontWeight: FontWeight.*bold*,  
 ),  
 ),  
 style: ElevatedButton.*styleFrom*(  
 backgroundColor: Colors.*white*, // Background color  
 foregroundColor: Colors.*pink*.shade800, // Foreground color (for border, ripple, etc.)  
 side: BorderSide(  
 color: Colors.*pink*.shade800, // Border color  
 width: 2.0,  
 ),  
 shape: RoundedRectangleBorder(  
 borderRadius: BorderRadius.circular(30.0), // Rounded corners  
 ),  
 fixedSize: Size(220.0, 60.0), // Width and height  
 ),  
 ),  
 ),  
 ),  
 Padding(  
 padding: const EdgeInsets.all(8.0),  
 child: Container(  
 height: 60,  
 width: 220,  
 child: ElevatedButton.icon(  
 onPressed: () {  
 // Add your onPressed functionality here  
 },  
 icon: Icon(  
 Icons.*expand\_sharp*, // Choose any suitable icon  
 color: Colors.*pink*,  
 ),  
 label: Text(  
 'Other', // Replace with your desired title  
 style: TextStyle(  
 color: Colors.*pink*,  
 fontWeight: FontWeight.*bold* ),  
 ),  
 style: ElevatedButton.*styleFrom*(  
 backgroundColor: Colors.*white*, // Background color  
 foregroundColor: Colors.*pink*,  
 side: BorderSide(  
 color: Colors.*pink*, // Border color  
 width: 2.0,  
 )// Text and icon color  
 ),  
 ),  
 ),  
 ),  
 ],  
 ),  
 SizedBox(height: 20,),  
 Row(  
 children: [  
 SizedBox(width: 25,),  
 Container(  
 height: 30,  
 width: 100,  
 decoration: BoxDecoration(  
 borderRadius: BorderRadius.circular(15),  
 color: Colors.*pink*.shade900,  
 ),  
 child: Center(child: Text('MUN Stars',style: TextStyle(color: Colors.*white*,fontWeight: FontWeight.*bold*,fontSize: 14),)),  
 ),  
 SizedBox(width: 8,),  
 Text('16 Jan. 2025',style: TextStyle(color: Colors.*black*,fontSize: 14),)  
 ],  
 ),  
 SizedBox(height: 25,),  
 Row(  
 children: [  
 SizedBox(width: 25,),  
 CircleAvatar(  
 radius: 20,  
 backgroundImage: NetworkImage('https://mymun.com/uploads/f185c5d12cd516bb2b7301009edacb2101181624\_profile.webp'),  
 ),  
 SizedBox(width: 8,),  
 Text('Beatriz Santos\nMayo',style: TextStyle(color: Colors.*black*,fontWeight: FontWeight.*bold*,fontSize: 16),)  
 ],  
 ),  
 SizedBox(height: 15,),  
 Row(  
 children: [  
 SizedBox(width: 10,),  
 Text('“Being curious and open to learning new things\nis what will drive your success in MUN."',style: TextStyle(color: Colors.*blueGrey*,fontWeight: FontWeight.*bold*,fontSize: 22),)  
 ],  
 ),  
 SizedBox(height: 10,),  
 Row(  
 children: [  
 SizedBox(width: 10,),  
 Text('MUN Stars - Luna Ragozino',style: TextStyle(color: Colors.*blueGrey*,fontSize: 20),),  
 ],  
 ),  
 Container(  
 height: 400,  
 width: MediaQuery.*of*(context).size.width,  
 decoration: BoxDecoration(  
 borderRadius: BorderRadius.circular(15),  
 image: DecorationImage(image: NetworkImage('https://mymun.com/uploads/cac141bef8ffc43cfb48f918766abb3352bbd8d0.webp'))  
 ),  
 ),  
 Row(  
 children: [  
 SizedBox(width: 10,),  
 Text("“Bea, do you think that your level of Turkish\nis enough to chair in a MUN?” Jonas texted one day.\nThe answer is no, unfortunately, but that makes me\nthink. MUN in Turkish? What about in other\nlanguages, Italian, Spanish, French…? That's when I\nlearned about MedMUN, and consequently, thanks\nto Agnese, I met Luna, the Conference Director for\nMedMUN 2024. ",style: TextStyle(color: Colors.*black*,fontSize: 19,),)  
 ],  
 ),  
 SizedBox(height: 13,),  
 Row(  
 children: [  
 SizedBox(width: 7,),  
 TextButton.icon(onPressed: (){},  
 icon: Icon(Icons.*arrow\_forward\_outlined*,color: Colors.*yellow*.shade700,size: 20,),  
 label: Text('Read Full More',style: TextStyle(color: Colors.*yellow*.shade700,fontSize: 20),)),  
 ],  
 ),  
 SizedBox(height: 7,),  
 Divider(),  
 SizedBox(height: 10,),  
 Row(  
 children: [  
 SizedBox(width: 20,),  
 Container(  
 height: 30,  
 width: 100,  
 decoration: BoxDecoration(  
 color: Colors.*green*,  
 borderRadius: BorderRadius.circular(15)  
 ),  
 child: Center(  
 child: Text('Education',style: TextStyle(color: Colors.*white*,fontWeight: FontWeight.*bold*,fontSize: 14),),  
 ),  
 ),  
 SizedBox(width: 8,),  
 Text('12 Jan. 2025',style: TextStyle(color: Colors.*black*,fontSize: 14),)  
 ],  
 ),  
 SizedBox(height: 25,),  
 Row(  
 children: [  
 SizedBox(width: 25,),  
 CircleAvatar(  
 radius: 20,  
 backgroundImage: NetworkImage('https://mymun.com/uploads/f185c5d12cd516bb2b7301009edacb2101181624\_profile.webp'),  
 ),  
 SizedBox(width: 8,),  
 Text('Beatriz Santos Mayo',style: TextStyle(color: Colors.*black*,fontWeight: FontWeight.*bold*,fontSize: 16),)  
 ],  
 ),  
 SizedBox(height: 10,),  
 Row(  
 children: [  
 SizedBox(width: 10,),  
 Text('Manual to be a Social BEE',style: TextStyle(color: Colors.*blueGrey*,fontSize: 24,fontWeight: FontWeight.*bold*),)  
 ],  
 ),  
 SizedBox(height: 10,),  
 Row(  
 children: [  
 SizedBox(width: 10,),  
 Text('How to drive MUN socials without dying',style: TextStyle(color: Colors.*blueGrey*,fontSize: 20),)  
 ],  
 ),  
 SizedBox(height: 10,),  
 Container(  
 height: 400,  
 width: MediaQuery.*of*(context).size.height,  
 decoration: BoxDecoration(  
 borderRadius: BorderRadius.circular(15),  
 image: DecorationImage(image: NetworkImage('https://mymun.com/uploads/944356fd029bf03f35aa825995786bdc65cafa04.webp'))  
 ),  
 ),  
 SizedBox(height: 10,),  
 Row(  
 children: [  
 SizedBox(  
 width: 10,  
 ),  
 Text('Let’s be honest: we all know at least one person\nwho overslept after a wild night at a MUN Social.\nAnd if you don’t know anyone like that, sorry to tell\nyou, but it might be you. Personally, I’ve never been\nthat person (thank you very much) because if there’s\none thing I hate, it’s people coming to the\ncommittee session late. But don’t get\nmisunderstood; I’m not skipping the party\neither. I’m that social butterfly who knows how to\nhave fun and still show up ready to slay\nthe following day. If you’re vibing with that or want to\nkill it in your socials, follow these tips to crush th\n MUN nightlife.',style: TextStyle(color: Colors.*black*,fontSize: 19),)  
 ],  
 ),  
 SizedBox(height: 13,),  
 Row(  
 children: [  
 SizedBox(width: 7,),  
 TextButton.icon(onPressed: (){},  
 icon: Icon(Icons.*arrow\_forward\_outlined*,color: Colors.*yellow*.shade700,size: 20,),  
 label: Text('Read Full More',style: TextStyle(color: Colors.*yellow*.shade700,fontSize: 20),)),  
 ],  
 ),  
 SizedBox(height: 7,),  
 Divider(),  
 SizedBox(height: 10,),  
 Row(  
 children: [  
 SizedBox(width: 25,),  
 Container(  
 height: 30,  
 width: 100,  
 decoration: BoxDecoration(  
 borderRadius: BorderRadius.circular(15),  
 color: Colors.*pink*.shade900,  
 ),  
 child: Center(child: Text('MUN Stars',style: TextStyle(color: Colors.*white*,fontWeight: FontWeight.*bold*,fontSize: 14),)),  
 ),  
 SizedBox(width: 8,),  
 Text('16 Jan. 2025',style: TextStyle(color: Colors.*black*,fontSize: 14),)  
 ],  
 ),  
 SizedBox(height: 25,),  
 Row(  
 children: [  
 SizedBox(width: 25,),  
 CircleAvatar(  
 radius: 20,  
 backgroundImage: NetworkImage('https://mymun.com/uploads/f185c5d12cd516bb2b7301009edacb2101181624\_profile.webp'),  
 ),  
 SizedBox(width: 8,),  
 Text('Beatriz Santos Mayo',style: TextStyle(color: Colors.*black*,fontWeight: FontWeight.*bold*,fontSize: 16),)  
 ],  
 ),  
 SizedBox(height: 10,),  
 Row(  
 children: [  
 SizedBox(width: 20,),  
 Text('"MUN is about learning, growing,\nand realizing you’re capable of so\nmuch more than you think."',style: TextStyle(color: Colors.*blueGrey*,fontSize: 30,fontWeight: FontWeight.*bold*),)  
 ],  
 ),  
 SizedBox(height: 10,),  
 Row(  
 children: [  
 SizedBox(width: 20,),  
 Text('MUN Stars - Kristina Wiese',style: TextStyle(color: Colors.*blueGrey*,fontSize: 24),)  
 ],  
 ),  
 SizedBox(height: 10,),  
 Container(  
 height: 400,  
 width: MediaQuery.*of*(context).size.width,  
 decoration: BoxDecoration(  
 image: DecorationImage(image: NetworkImage('https://mymun.com/uploads/f21be509eddc521874bbb6aaef0357eb8bf7ad03.webp'))  
 ),  
 ),  
 Row(  
 children: [  
 SizedBox(width: 20,),  
 Text('I met Kristina when I chaired her students at\nKAMUN 2024. She was, honestly, everyone’s\nfavourite teacher: a mix of energy, warmth,\nand genuine care for her students. ',style: TextStyle(color: Colors.*black*,fontSize: 22),)  
 ],  
 ),  
 SizedBox(height: 13,),  
 Row(  
 children: [  
 SizedBox(width: 7,),  
 TextButton.icon(onPressed: (){},  
 icon: Icon(Icons.*arrow\_forward\_outlined*,color: Colors.*yellow*.shade700,size: 20,),  
 label: Text('Read Full More',style: TextStyle(color: Colors.*yellow*.shade700,fontSize: 20),)),  
 ],  
 ),  
 SizedBox(height: 7,),  
 Divider(),  
 SizedBox(height: 10,),  
 Row(  
 children: [  
 SizedBox(width: 25,),  
 Container(  
 height: 30,  
 width: 100,  
 decoration: BoxDecoration(  
 borderRadius: BorderRadius.circular(15),  
 color: Colors.*pink*.shade900,  
 ),  
 child: Center(child: Text('MUN Stars',style: TextStyle(color: Colors.*white*,fontWeight: FontWeight.*bold*,fontSize: 14),)),  
 ),  
 SizedBox(width: 8,),  
 Text('16 Jan. 2025',style: TextStyle(color: Colors.*black*,fontSize: 14),)  
 ],  
 ),  
 SizedBox(height: 25,),  
 Row(  
 children: [  
 SizedBox(width: 25,),  
 CircleAvatar(  
 radius: 20,  
 backgroundImage: NetworkImage('https://mymun.com/uploads/f185c5d12cd516bb2b7301009edacb2101181624\_profile.webp'),  
 ),  
 SizedBox(width: 8,),  
 Text('Beatriz Santos Mayo',style: TextStyle(color: Colors.*black*,fontWeight: FontWeight.*bold*,fontSize: 16),)  
 ],  
 ),  
 SizedBox(height: 10,),  
 Row(  
 children: [  
 SizedBox(width: 20,),  
 Text('“Work hard and play hard is my\nmain advice."',style: TextStyle(color: Colors.*blueGrey*,fontWeight: FontWeight.*bold*,fontSize: 30),),  
 ],  
 ),  
 SizedBox(height: 10,),  
 Row(  
 children: [  
 SizedBox(width: 20,),  
 Text('MUN Stars - Adriana Lucero Dorst',style: TextStyle(color: Colors.*blueGrey*,fontSize: 24),)  
 ],  
 ),  
 SizedBox(height: 10,),  
 Container(  
 height: 400,  
 width: MediaQuery.*of*(context).size.width,  
 decoration: BoxDecoration(  
 image: DecorationImage(image: NetworkImage('https://mymun.com/uploads/2df2d63d3d39c54da1f50b93d43fc5ce10c9495b.webp'))  
 ),  
 ),  
 Row(  
 children: [  
 SizedBox(width: 20,),  
 Text("Sometimes networks don’t go that far—or do they?\nOne sunny afternoon in Leuven, I was walking with\nmy friend Jose, who I met through our mutual MUN\nfriend Marie-Sophie (sending lots of love if you’re\nreading this!). We talked about this interview\nsection, and I asked if he’d be interested in being\nfeatured. Jose, ever the modest one, laughed\nand said he was “too old for this” and had officially\nretired from MUN. But, knowing Jose, I’m not so\nsure—he still shows up to conferences! Instead,he\nsuggested interviewing Adriana. “You’ll love her,” he\nsaid, and he wasn’t wrong.",style: TextStyle(color: Colors.*black*,fontSize: 20),)  
 ],  
 ),  
 SizedBox(height: 13,),  
 Row(  
 children: [  
 SizedBox(width: 7,),  
 TextButton.icon(onPressed: (){},  
 icon: Icon(Icons.*arrow\_forward\_outlined*,color: Colors.*yellow*.shade700,size: 20,),  
 label: Text('Read Full More',style: TextStyle(color: Colors.*yellow*.shade700,fontSize: 20),)),  
 ],  
 ),  
 SizedBox(height: 7,),  
 Divider(),  
 SizedBox(height: 40,),  
 Container(  
 height: 80,  
 width: MediaQuery.*of*(context).size.width,  
 child: Row(  
 children: [  
 IconButton(  
 onPressed: (){},  
 icon: Icon(FontAwesomeIcons.*tiktok*,color: Colors.*blueGrey*,)  
 ),  
 SizedBox(width: 3,),IconButton(  
 onPressed: (){},  
 icon: Icon(FontAwesomeIcons.*instagram*,color: Colors.*blueGrey*,)  
 ),  
 SizedBox(width: 3,),  
 IconButton(onPressed: (){}, icon: Icon(FontAwesomeIcons.*discord*,color: Colors.*blueGrey*,)),  
 SizedBox(width: 3,),IconButton(  
 onPressed: (){},  
 icon: Icon(FontAwesomeIcons.*linkedin*,color: Colors.*blueGrey*,)  
 ),  
 SizedBox(width: 3,),IconButton(  
 onPressed: (){},  
 icon: Icon(FontAwesomeIcons.*spotify*,color: Colors.*blueGrey*,)  
 ),  
 SizedBox(width: 218,),  
 TextButton(onPressed: (){}, child: Text('Imprint',style: TextStyle(color: Colors.*blueGrey*),))  
 ],  
 ),  
 )  
 ],  
 ),  
 ),  
 ),  
 );  
 }  
}

//CHAT SCREEN DART

// import 'package:flutter/material.dart';  
// import 'api\_service.dart';  
// import 'custom\_sidebar.dart'; // Import the custom sidebar  
//  
// class ChatScreen extends StatefulWidget {  
// const ChatScreen({super.key});  
//  
// @override  
// \_ChatScreenState createState() => \_ChatScreenState();  
// }  
//  
// class \_ChatScreenState extends State<ChatScreen> {  
// final TextEditingController \_controller = TextEditingController();  
// final List<Map<String, String>> \_messages = [];  
// final GlobalKey<ScaffoldState> \_scaffoldKey = GlobalKey<ScaffoldState>();  
//  
// void \_sendMessage() async {  
// final userInput = \_controller.text;  
// if (userInput.isEmpty) return;  
//  
// setState(() {  
// \_messages.add({'user': userInput});  
// \_controller.clear();  
// });  
//  
// final response = await ApiService.getBotResponse(userInput);  
//  
// setState(() {  
// \_messages.add({'bot': response});  
// });  
// }  
//  
// @override  
// Widget build(BuildContext context) {  
// return Scaffold(  
// key: \_scaffoldKey,  
// drawer: const CustomSidebar(),  
// appBar: AppBar(  
// title: const Text(  
// 'MyMun Chatbot',  
// style: TextStyle(  
// fontWeight: FontWeight.bold,  
// color: Colors.white,  
// fontSize: 20,  
// ),  
// ),  
// centerTitle: true,  
// backgroundColor: Colors.black,  
// leading: IconButton(  
// icon: const Icon(Icons.menu, color: Colors.white),  
// onPressed: () => \_scaffoldKey.currentState?.openDrawer(),  
// )  
// ),  
// body: Column(  
// children: [  
// Expanded(  
// child: Container(  
// color: Colors.black,  
// child: ListView.builder(  
// padding: const EdgeInsets.symmetric(horizontal: 15, vertical: 10),  
// itemCount: \_messages.length,  
// itemBuilder: (context, index) {  
// final message = \_messages[index];  
// final isUser = message.containsKey('user');  
// return Align(  
// alignment: isUser ? Alignment.centerRight : Alignment.centerLeft,  
// child: Container(  
// margin: const EdgeInsets.symmetric(vertical: 8),  
// padding: const EdgeInsets.all(15),  
// decoration: BoxDecoration(  
// color: isUser ? Colors.white : Colors.grey[900],  
// borderRadius: BorderRadius.only(  
// topLeft: Radius.circular(isUser ? 12 : 0),  
// topRight: Radius.circular(isUser ? 0 : 12),  
// bottomLeft: const Radius.circular(12),  
// bottomRight: const Radius.circular(12),  
// ),  
// ),  
// child: Text(  
// isUser ? message['user']! : message['bot']!,  
// style: TextStyle(  
// fontSize: 16,  
// fontWeight: FontWeight.w500,  
// color: isUser ? Colors.black : Colors.white,  
// ),  
// ),  
// ),  
// );  
// },  
// ),  
// ),  
// ),  
// Container(  
// padding: const EdgeInsets.all(10.0),  
// decoration: BoxDecoration(  
// color: Colors.grey[900],  
// boxShadow: [  
// BoxShadow(  
// color: Colors.grey.withOpacity(0.2),  
// spreadRadius: 1,  
// blurRadius: 5,  
// ),  
// ],  
// ),  
// child: Row(  
// children: [  
// Expanded(  
// child: TextField(  
// controller: \_controller,  
// style: const TextStyle(color: Colors.white),  
// decoration: InputDecoration(  
// hintText: 'Type your message...',  
// hintStyle: TextStyle(color: Colors.grey[500]),  
// border: OutlineInputBorder(  
// borderRadius: BorderRadius.circular(10),  
// borderSide: BorderSide.none,  
// ),  
// filled: true,  
// fillColor: Colors.grey[800],  
// ),  
// ),  
// ),  
// const SizedBox(width: 8),  
// FloatingActionButton(  
// onPressed: \_sendMessage,  
// child: const Icon(Icons.send, color: Colors.black),  
// backgroundColor: Colors.white,  
// ),  
// ],  
// ),  
// ),  
// ],  
// ),  
// );  
// }  
// }  
  
import 'package:chat\_gpt\_sdk/chat\_gpt\_sdk.dart';  
import 'package:dash\_chat\_2/dash\_chat\_2.dart';  
import 'package:flutter/material.dart';  
import 'package:mohsin/custom\_sidebar.dart';  
  
class ChatScreen extends StatefulWidget {  
 const ChatScreen({super.key});  
  
 @override  
 \_ChatScreenState createState() => \_ChatScreenState();  
}  
  
class \_ChatScreenState extends State<ChatScreen> {  
 final TextEditingController \_controller = TextEditingController();  
 final List<ChatMessage> \_messages = [];  
 final GlobalKey<ScaffoldState> \_scaffoldKey = GlobalKey<ScaffoldState>();  
  
 OpenAI \_openAi = OpenAI.*instance*.build(  
 token: "sk-svcacct-EpIaCMm78nUoNRf4wxHGJX6CiBhHOXhOppjQmPRCU1XwjpCIDPv\_gnazYvz6B28T3BlbkFJkaTEjQLl6PRLc0XNj7tkFeUCcUU5C\_EJ4uBn8NPomstzz2Y-Xf55XTStLrCGtbgA", // Replace with secure token handling  
 baseOption: HttpSetup(receiveTimeout: const Duration(seconds: 10)),  
 enableLog: true,  
 );  
  
 final ChatUser user = ChatUser(  
 id: '1',  
 firstName: 'Charles',  
 lastName: 'Leclerc',  
 );  
  
 final ChatUser gpt = ChatUser(  
 id: '2',  
 firstName: 'Chat',  
 lastName: 'GPT',  
 );  
  
 @override  
 void initState() {  
 super.initState();  
 }  
  
 void \_sendMessage() async {  
 final userInput = \_controller.text.trim();  
 if (userInput.isEmpty) return;  
  
 final userMessage = ChatMessage(  
 user: user,  
 text: userInput,  
 createdAt: DateTime.now(),  
 );  
  
 setState(() {  
 \_messages.add(userMessage);  
 \_controller.clear();  
 });  
  
 await \_getChatResponse(userInput);  
 }  
  
 Future<void> \_getChatResponse(String userInput) async {  
 try {  
 final request = ChatCompleteText(  
 model: Gpt4oMiniChatModel(),  
 messages: \_messages.map((msg) {  
 return {  
 "role": msg.user.id == user.id ? "user" : "assistant", // Correct role assignment  
 "content": msg.text,  
 };  
 }).toList(),  
 maxToken: 800,  
 );  
  
 final response = await \_openAi.onChatCompletion(request: request);  
  
 if (response != null && response.choices.isNotEmpty) {  
 final botMessage = ChatMessage(  
 user: gpt,  
 text: response.choices.first.message?.content ?? "No response",  
 createdAt: DateTime.now(),  
 );  
  
 setState(() {  
 \_messages.add(botMessage);  
 });  
 }  
 } catch (e) {  
 final errorMessage = ChatMessage(  
 user: gpt,  
 text: '$e',  
 createdAt: DateTime.now(),  
 );  
 print(e);  
 setState(() {  
 \_messages.add(errorMessage);  
 });  
 }  
 }  
  
 @override  
 Widget build(BuildContext context) {  
 return Scaffold(  
 drawer: const CustomSidebar(), key: \_scaffoldKey,  
 appBar: AppBar(  
 title: const Text(  
 'MyMun Chatbot',  
 style: TextStyle(  
 fontWeight: FontWeight.*bold*,  
 color: Colors.*white*,  
 fontSize: 20,  
 ),  
 ),  
 centerTitle: true,  
 backgroundColor: Colors.*black*,  
  
 ),  
 body: Column(  
 children: [  
 Expanded(  
 child: Container(  
 color: Colors.*black*,  
 child: ListView.builder(  
 padding: const EdgeInsets.symmetric(horizontal: 15, vertical: 10),  
 itemCount: \_messages.length,  
 itemBuilder: (context, index) {  
 final message = \_messages[index];  
 final isUser = message.user.id == user.id;  
 return Align(  
 alignment: isUser ? Alignment.*centerRight* : Alignment.*centerLeft*,  
 child: Container(  
 margin: const EdgeInsets.symmetric(vertical: 8),  
 padding: const EdgeInsets.all(15),  
 decoration: BoxDecoration(  
 color: isUser ? Colors.*white* : Colors.*grey*[900],  
 borderRadius: BorderRadius.only(  
 topLeft: Radius.circular(isUser ? 12 : 0),  
 topRight: Radius.circular(isUser ? 0 : 12),  
 bottomLeft: const Radius.circular(12),  
 bottomRight: const Radius.circular(12),  
 ),  
 ),  
 child: Text(  
 message.text,  
 style: TextStyle(  
 fontSize: 16,  
 fontWeight: FontWeight.*w500*,  
 color: isUser ? Colors.*black* : Colors.*white*,  
 ),  
 ),  
 ),  
 );  
 },  
 ),  
 ),  
 ),  
 Container(  
 padding: const EdgeInsets.all(10.0),  
 decoration: BoxDecoration(  
 color: Colors.*grey*[900],  
 boxShadow: [  
 BoxShadow(  
 color: Colors.*grey*.withOpacity(0.2),  
 spreadRadius: 1,  
 blurRadius: 5,  
 ),  
 ],  
 ),  
 child: Row(  
 children: [  
 Expanded(  
 child: TextField(  
 controller: \_controller,  
 style: const TextStyle(color: Colors.*white*),  
 decoration: InputDecoration(  
 hintText: 'Type your message...',  
 hintStyle: TextStyle(color: Colors.*grey*[500]),  
 border: OutlineInputBorder(  
 borderRadius: BorderRadius.circular(10),  
 borderSide: BorderSide.*none*,  
 ),  
 filled: true,  
 fillColor: Colors.*grey*[800],  
 ),  
 ),  
 ),  
 const SizedBox(width: 8),  
 FloatingActionButton(  
 onPressed: \_sendMessage,  
 child: const Icon(Icons.*send*, color: Colors.*black*),  
 backgroundColor: Colors.*white*,  
 ),  
 ],  
 ),  
 ),  
 ],  
 ),  
 );  
 }  
}

//CUSTOME SIDEBAR DART

import 'package:firebase\_auth/firebase\_auth.dart';  
import 'package:flutter/material.dart';  
import 'package:mohsin/authentication/authstream.dart';  
import 'package:mohsin/authentication/login.dart';  
import 'authentication/cheat\_sheet.dart';  
import 'blogs.dart';  
import 'payment.dart';  
import 'profile.dart';  
  
class CustomSidebar extends StatelessWidget {  
 final String? userEmail;  
 final String? userName;  
  
 const CustomSidebar({  
 Key? key,  
 this.userEmail,  
 this.userName,  
 }) : super(key: key);  
  
 @override  
 Widget build(BuildContext context) {  
 final User? user = FirebaseAuth.*instance*.currentUser; // Fetch current user  
  
 final String userName = user?.displayName ?? 'User Name'; // Fallback if displayName is null  
 final String userEmail = user?.email ?? 'user@example.com';  
 return Drawer(  
 backgroundColor: Colors.*white*,  
 child: ListView(  
 padding: EdgeInsets.*zero*,  
 children: [  
 DrawerHeader(  
 decoration: const BoxDecoration(  
 color: Colors.*black*,  
 ),  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.start,  
 children: [  
 const CircleAvatar(  
 radius: 30,  
 backgroundColor: Colors.*white*,  
 child: Icon(  
 Icons.*person*,  
 size: 35,  
 color: Colors.*black*,  
 ),  
 ),  
 const SizedBox(height: 10),  
 Text(  
 userName ?? 'User Name',  
 style: const TextStyle(  
 color: Colors.*white*,  
 fontSize: 18,  
 fontWeight: FontWeight.*bold*,  
 ),  
 ),  
 Text(  
 userEmail ?? 'user@example.com',  
 style: const TextStyle(  
 color: Colors.*white70*,  
 fontSize: 14,  
 ),  
 ),  
 ],  
 ),  
 ),  
 \_buildMenuItem(  
 context,  
 icon: Icons.*chat*,  
 title: 'ChatBot',  
 onTap: () => Navigator.*pushReplacementNamed*(context, '/'),  
 ),  
 \_buildMenuItem(  
 context,  
 icon: Icons.*event*,  
 title: 'Register a Conference',  
 onTap: () => Navigator.*pushNamed*(context, '/conferences'),  
 ),  
 \_buildMenuItem(  
 context,  
 icon: Icons.*description*,  
 title: "Cheat Sheet",  
 onTap: (){  
 Navigator.*push*(context, MaterialPageRoute(builder: (context)=>CheatSheet()));  
 }  
 ),  
 \_buildMenuItem(  
 context,  
 icon: Icons.*article*,  
 title: 'Blogs',  
 onTap: (){  
 Navigator.*push*(context, MaterialPageRoute(builder: (context)=>Blogs()));  
 }  
 ),  
 \_buildMenuItem(  
 context,  
 icon: Icons.*person\_outline*,  
 title: 'Profile',  
 onTap:(){  
 Navigator.*push*(  
 context,  
 MaterialPageRoute(  
 builder: (context) => ProfileScreen(  
 userEmail: userEmail,  
 userName: userName ?? 'User Name',  
 ),  
 ),  
 );  
 },  
 ),  
 const Divider(color: Colors.*grey*),  
 \_buildMenuItem(  
 context,  
 icon: Icons.*logout*,  
 title: 'Logout',  
 onTap: ()  
 {  
 FirebaseAuth \_auth = FirebaseAuth.*instance*;  
 \_auth.signOut();  
 Navigator.*push*(context, MaterialPageRoute(builder: (context) => Authstream()));  
 }  
 ),  
 ],  
 ),  
 );  
 }  
  
 Widget \_buildMenuItem(  
 BuildContext context, {  
 required IconData icon,  
 required String title,  
 required VoidCallback onTap,  
 }) {  
 return ListTile(  
 leading: Icon(icon, color: Colors.*black*),  
 title: Text(  
 title,  
 style: const TextStyle(  
 color: Colors.*black*,  
 fontSize: 16,  
 ),  
 ),  
 onTap: () {  
 Navigator.*pop*(context); // Close drawer  
 onTap();  
 },  
 hoverColor: Colors.*grey*[200],  
 );  
 }  
}

//FIRE BASE DART

// File generated by FlutterFire CLI.  
// ignore\_for\_file: type=lint  
import 'package:firebase\_core/firebase\_core.dart' show FirebaseOptions;  
import 'package:flutter/foundation.dart'  
 show defaultTargetPlatform, kIsWeb, TargetPlatform;  
  
*/// Default [FirebaseOptions] for use with your Firebase apps.  
///  
/// Example:  
/// ```dart  
/// import 'firebase\_options.dart';  
/// // ...  
/// await Firebase.initializeApp(  
/// options: DefaultFirebaseOptions.currentPlatform,  
/// );  
/// ```*class DefaultFirebaseOptions {  
 static FirebaseOptions get *currentPlatform* {  
 if (kIsWeb) {  
 return *web*;  
 }  
 switch (defaultTargetPlatform) {  
 case TargetPlatform.android:  
 return *android*;  
 case TargetPlatform.iOS:  
 throw UnsupportedError(  
 'DefaultFirebaseOptions have not been configured for ios - '  
 'you can reconfigure this by running the FlutterFire CLI again.',  
 );  
 case TargetPlatform.macOS:  
 throw UnsupportedError(  
 'DefaultFirebaseOptions have not been configured for macos - '  
 'you can reconfigure this by running the FlutterFire CLI again.',  
 );  
 case TargetPlatform.windows:  
 throw UnsupportedError(  
 'DefaultFirebaseOptions have not been configured for windows - '  
 'you can reconfigure this by running the FlutterFire CLI again.',  
 );  
 case TargetPlatform.linux:  
 throw UnsupportedError(  
 'DefaultFirebaseOptions have not been configured for linux - '  
 'you can reconfigure this by running the FlutterFire CLI again.',  
 );  
 default:  
 throw UnsupportedError(  
 'DefaultFirebaseOptions are not supported for this platform.',  
 );  
 }  
 }  
  
 static const FirebaseOptions *web* = FirebaseOptions(  
 apiKey: 'AIzaSyC6\_K6NzjkhEroJoMepJirYyCNN51NjhXg',  
 appId: '1:949741847240:web:b9860d579e657bf3839bfd',  
 messagingSenderId: '949741847240',  
 projectId: 'semester-project-db23a',  
 authDomain: 'semester-project-db23a.firebaseapp.com',  
 storageBucket: 'semester-project-db23a.firebasestorage.app',  
 measurementId: 'G-LR5NX4WNCY',  
 );  
  
 static const FirebaseOptions *android* = FirebaseOptions(  
 apiKey: 'AIzaSyCxQ-Wp993O-wnId3BxNzE9Lbn98EZqsIA',  
 appId: '1:949741847240:android:cafe538a016490c0839bfd',  
 messagingSenderId: '949741847240',  
 projectId: 'semester-project-db23a',  
 storageBucket: 'semester-project-db23a.firebasestorage.app',  
 );  
}

//CONFERENCE SCREEN DART

import 'package:flutter/material.dart';  
import 'package:cloud\_firestore/cloud\_firestore.dart';  
import 'package:firebase\_core/firebase\_core.dart';  
import 'package:mohsin/payment.dart';  
  
class ConferenceScreen extends StatefulWidget {  
 @override  
 \_ConferenceScreenState createState() => \_ConferenceScreenState();  
}  
  
class \_ConferenceScreenState extends State<ConferenceScreen> {  
 final FirebaseFirestore \_firestore = FirebaseFirestore.*instance*;  
  
 String selectedType = 'All';  
 String selectedCountry = 'All';  
 String searchQuery = '';  
  
 List<String> conferenceTypes = ['All', 'MUN', 'Debate', 'Tech'];  
 List<String> countries = ['All', 'USA', 'UK'];  
  
 Future<void> \_deleteConference(String id) async {  
 await \_firestore.collection('conferences').doc(id).delete();  
 }  
  
 void \_showConferenceDetails(BuildContext context, Map<String, dynamic> conference) {  
 showDialog(  
 context: context,  
 builder: (context) => Dialog(  
 shape: RoundedRectangleBorder(  
 borderRadius: BorderRadius.circular(12),  
 ),  
 child: SingleChildScrollView(  
 child: Column(  
 mainAxisSize: MainAxisSize.min,  
 children: [  
 ClipRRect(  
 borderRadius: const BorderRadius.vertical(top: Radius.circular(12)),  
 child: Image.network(  
 conference['image']!,  
 height: 200,  
 width: double.*infinity*,  
 fit: BoxFit.cover,  
 errorBuilder: (context, error, stackTrace) {  
 return Container(  
 height: 200,  
 color: Colors.*grey*[300],  
 child: Center(  
 child: Icon(  
 Icons.*error\_outline*,  
 size: 50,  
 color: Colors.*grey*[600],  
 ),  
 ),  
 );  
 },  
 ),  
 ),  
 Padding(  
 padding: const EdgeInsets.all(16.0),  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.start,  
 children: [  
 Text(  
 conference['title']!,  
 style: const TextStyle(fontSize: 20, fontWeight: FontWeight.*bold*),  
 ),  
 const SizedBox(height: 10),  
 \_detailRow(Icons.*location\_on*, 'Location', conference['location']!),  
 \_detailRow(Icons.*calendar\_today*, 'Date', conference['date']!),  
 \_detailRow(Icons.*category*, 'Type', conference['type']!),  
 \_detailRow(Icons.*timer*, 'Application Deadline', conference['applicationDeadline']!),  
 \_detailRow(Icons.*attach\_money*, 'Fees', conference['fees']!),  
 ],  
 ),  
 ),  
 Padding(  
 padding: const EdgeInsets.all(16.0),  
 child: ElevatedButton(  
 onPressed: () {  
 Navigator.*pop*(context); // Close the dialog  
 Navigator.*push*(  
 context,  
 MaterialPageRoute(  
 builder: (context) => PaymentScreen(conference: conference),  
 ),  
 );  
 },  
 style: ElevatedButton.*styleFrom*(  
 backgroundColor: Colors.*black*,  
 padding: const EdgeInsets.symmetric(vertical: 16),  
 ),  
 child: const Text('Register', style: TextStyle(color: Colors.*white*)),  
 ),  
 ),  
 ],  
 ),  
 ),  
 ),  
 );  
 }  
  
 Widget \_detailRow(IconData icon, String label, String value) {  
 return Padding(  
 padding: const EdgeInsets.symmetric(vertical: 4),  
 child: Row(  
 children: [  
 Icon(icon, size: 20, color: Colors.*grey*[600]),  
 const SizedBox(width: 8),  
 Text('$label: $value', style: const TextStyle(fontSize: 14)),  
 ],  
 ),  
 );  
 }  
  
 @override  
 Widget build(BuildContext context) {  
 return Scaffold(  
 backgroundColor: Colors.*grey*[100],  
 appBar: AppBar(  
 title: const Text(  
 'CONFERENCES',  
 style: TextStyle(  
 color: Colors.*white*,  
 fontWeight: FontWeight.*bold*,  
 letterSpacing: 1.5,  
 ),  
 ),  
 centerTitle: true,  
 backgroundColor: Colors.*black*,  
 elevation: 0,  
 ),  
 floatingActionButton: FloatingActionButton(  
 backgroundColor: Colors.*black*,  
 onPressed: () async {  
 final newConference = await Navigator.*push*(  
 context,  
 MaterialPageRoute(  
 builder: (context) => AddConferenceScreen(),  
 ),  
 );  
 if (newConference != null) {  
 await \_firestore.collection('conferences').add(newConference);  
 }  
 },  
 child: const Icon(Icons.*add*, color: Colors.*white*),  
 ),  
 body: Column(  
 children: [  
 Container(  
 color: Colors.*black*,  
 padding: const EdgeInsets.all(16.0),  
 child: Column(  
 children: [  
 TextField(  
 style: const TextStyle(color: Colors.*black*),  
 onChanged: (value) {  
 setState(() {  
 searchQuery = value;  
 });  
 },  
 decoration: InputDecoration(  
 hintText: 'Search conferences...',  
 hintStyle: TextStyle(color: Colors.*grey*[600]),  
 filled: true,  
 fillColor: Colors.*white*,  
 prefixIcon: const Icon(Icons.*search*, color: Colors.*black*),  
 border: OutlineInputBorder(  
 borderRadius: BorderRadius.circular(30),  
 borderSide: BorderSide.*none*,  
 ),  
 ),  
 ),  
 const SizedBox(height: 16),  
 Row(  
 children: [  
 Expanded(  
 child: DropdownButtonFormField<String>(  
 value: selectedType,  
 decoration: InputDecoration(  
 contentPadding: const EdgeInsets.symmetric(horizontal: 16),  
 filled: true,  
 fillColor: Colors.*white*,  
 border: OutlineInputBorder(  
 borderRadius: BorderRadius.circular(30),  
 borderSide: BorderSide.*none*,  
 ),  
 ),  
 items: conferenceTypes.map((type) {  
 return DropdownMenuItem(  
 value: type,  
 child: Text(type),  
 );  
 }).toList(),  
 onChanged: (value) {  
 setState(() {  
 selectedType = value!;  
 });  
 },  
 ),  
 ),  
 const SizedBox(width: 16),  
 Expanded(  
 child: DropdownButtonFormField<String>(  
 value: selectedCountry,  
 decoration: InputDecoration(  
 contentPadding: const EdgeInsets.symmetric(horizontal: 16),  
 filled: true,  
 fillColor: Colors.*white*,  
 border: OutlineInputBorder(  
 borderRadius: BorderRadius.circular(30),  
 borderSide: BorderSide.*none*,  
 ),  
 ),  
 items: countries.map((country) {  
 return DropdownMenuItem(  
 value: country,  
 child: Text(country),  
 );  
 }).toList(),  
 onChanged: (value) {  
 setState(() {  
 selectedCountry = value!;  
 });  
 },  
 ),  
 ),  
 ],  
 ),  
 ],  
 ),  
 ),  
 Expanded(  
 child: StreamBuilder<QuerySnapshot>(  
 stream: \_firestore.collection('conferences').snapshots(),  
 builder: (context, snapshot) {  
 if (!snapshot.hasData) {  
 return const Center(child: CircularProgressIndicator());  
 }  
  
 final documents = snapshot.data!.docs;  
 final filteredConferences = documents.where((doc) {  
 final data = doc.data() as Map<String, dynamic>;  
 if (selectedType != 'All' && data['type'] != selectedType) return false;  
 if (selectedCountry != 'All' && !data['location']!.contains(selectedCountry)) {  
 return false;  
 }  
 if (searchQuery.isNotEmpty &&  
 !data['title']!.toLowerCase().contains(searchQuery.toLowerCase())) {  
 return false;  
 }  
 return true;  
 }).toList();  
  
 return ListView.builder(  
 padding: const EdgeInsets.all(16),  
 itemCount: filteredConferences.length,  
 itemBuilder: (context, index) {  
 final document = filteredConferences[index];  
 final conference = document.data() as Map<String, dynamic>;  
  
 return Card(  
 elevation: 4,  
 margin: const EdgeInsets.only(bottom: 16),  
 shape: RoundedRectangleBorder(  
 borderRadius: BorderRadius.circular(12),  
 ),  
 child: InkWell(  
 onTap: () => \_showConferenceDetails(context, conference),  
 child: Stack(  
 children: [  
 Column(  
 crossAxisAlignment: CrossAxisAlignment.start,  
 children: [  
 ClipRRect(  
 borderRadius: const BorderRadius.vertical(  
 top: Radius.circular(12),  
 ),  
 child: Image.network(  
 conference['image']!,  
 height: 200,  
 width: double.*infinity*,  
 fit: BoxFit.cover,  
 errorBuilder: (context, error, stackTrace) {  
 return Container(  
 height: 200,  
 color: Colors.*grey*[300],  
 child: const Icon(Icons.*error*),  
 );  
 },  
 ),  
 ),  
 Container(  
 padding: const EdgeInsets.all(16),  
 child: Text(  
 conference['title']!,  
 style: const TextStyle(  
 fontSize: 20,  
 fontWeight: FontWeight.*bold*,  
 ),  
 ),  
 ),  
 ],  
 ),  
 Positioned(  
 top: 8,  
 right: 8,  
 child: IconButton(  
 icon: const Icon(Icons.*delete*, color: Colors.*red*),  
 onPressed: () => \_deleteConference(document.id),  
 ),  
 ),  
 ],  
 ),  
 ),  
 );  
 },  
 );  
 },  
 ),  
 ),  
 ],  
 ),  
 );  
 }  
}  
  
class AddConferenceScreen extends StatefulWidget {  
 @override  
 \_AddConferenceScreenState createState() => \_AddConferenceScreenState();  
}  
  
class \_AddConferenceScreenState extends State<AddConferenceScreen> {  
 final \_formKey = GlobalKey<FormState>();  
 final Map<String, dynamic> \_newConference = {  
 'title': '',  
 'location': '',  
 'date': '',  
 'type': 'MUN',  
 'applicationDeadline': '',  
 'fees': '',  
 'image': '',  
 };  
  
 final List<String> \_conferenceTypes = ['MUN', 'Debate', 'Tech'];  
  
 @override  
 Widget build(BuildContext context) {  
 return Scaffold(  
 appBar: AppBar(  
 title: const Text('Add Conference'),  
 backgroundColor: Colors.*black*,  
 ),  
 body: Padding(  
 padding: const EdgeInsets.all(16),  
 child: Form(  
 key: \_formKey,  
 child: ListView(  
 children: [  
 TextFormField(  
 decoration: const InputDecoration(  
 labelText: 'Title',  
 border: OutlineInputBorder(),  
 ),  
 validator: (value) {  
 if (value == null || value.isEmpty) {  
 return 'Please enter a title';  
 }  
 return null;  
 },  
 onSaved: (value) => \_newConference['title'] = value!,  
 ),  
 const SizedBox(height: 16),  
 TextFormField(  
 decoration: const InputDecoration(  
 labelText: 'Location',  
 border: OutlineInputBorder(),  
 ),  
 validator: (value) {  
 if (value == null || value.isEmpty) {  
 return 'Please enter a location';  
 }  
 return null;  
 },  
 onSaved: (value) => \_newConference['location'] = value!,  
 ),  
 const SizedBox(height: 16),  
 TextFormField(  
 decoration: const InputDecoration(  
 labelText: 'Date',  
 border: OutlineInputBorder(),  
 ),  
 keyboardType: TextInputType.*datetime*,  
 validator: (value) {  
 if (value == null || value.isEmpty) {  
 return 'Please enter a date';  
 }  
 return null;  
 },  
 onSaved: (value) => \_newConference['date'] = value!,  
 ),  
 const SizedBox(height: 16),  
 DropdownButtonFormField<String>(  
 value: \_newConference['type'],  
 decoration: const InputDecoration(  
 labelText: 'Type',  
 border: OutlineInputBorder(),  
 ),  
 items: \_conferenceTypes.map((type) {  
 return DropdownMenuItem(  
 value: type,  
 child: Text(type),  
 );  
 }).toList(),  
 onChanged: (value) {  
 setState(() {  
 \_newConference['type'] = value!;  
 });  
 },  
 ),  
 const SizedBox(height: 16),  
 TextFormField(  
 decoration: const InputDecoration(  
 labelText: 'Application Deadline',  
 border: OutlineInputBorder(),  
 ),  
 keyboardType: TextInputType.*datetime*,  
 validator: (value) {  
 if (value == null || value.isEmpty) {  
 return 'Please enter an application deadline';  
 }  
 return null;  
 },  
 onSaved: (value) => \_newConference['applicationDeadline'] = value!,  
 ),  
 const SizedBox(height: 16),  
 TextFormField(  
 decoration: const InputDecoration(  
 labelText: 'Fees',  
 border: OutlineInputBorder(),  
 ),  
 keyboardType: TextInputType.*number*,  
 validator: (value) {  
 if (value == null || value.isEmpty) {  
 return 'Please enter the fees';  
 }  
 return null;  
 },  
 onSaved: (value) => \_newConference['fees'] = value!,  
 ),  
 const SizedBox(height: 16),  
 TextFormField(  
 decoration: const InputDecoration(  
 labelText: 'Image URL',  
 border: OutlineInputBorder(),  
 ),  
 validator: (value) {  
 if (value == null || value.isEmpty) {  
 return 'Please enter an image URL';  
 }  
 return null;  
 },  
 onSaved: (value) => \_newConference['image'] = value!,  
 ),  
 const SizedBox(height: 24),  
 ElevatedButton(  
 onPressed: () {  
 if (\_formKey.currentState!.validate()) {  
 \_formKey.currentState!.save();  
 Navigator.*pop*(context, \_newConference);  
 }  
 },  
 style: ElevatedButton.*styleFrom*(  
 backgroundColor: Colors.*black*,  
 padding: const EdgeInsets.symmetric(vertical: 16),  
 shape: RoundedRectangleBorder(  
 borderRadius: BorderRadius.circular(8),  
 ),  
 ),  
 child: const Text('Add Conference'),  
 ),  
 ],  
 ),  
 ),  
 ),  
 );  
 }  
}

//PAYMENT DART

import 'package:flutter/material.dart';  
import 'package:crypto/crypto.dart';  
import 'package:http/http.dart' as http;  
import 'dart:convert';  
  
class PaymentScreen extends StatefulWidget {  
 final Map<String, dynamic> conference;  
  
 const PaymentScreen({Key? key, required this.conference}) : super(key: key);  
  
 @override  
 State createState() => \_PaymentScreenState();  
}  
  
class \_PaymentScreenState extends State<PaymentScreen> {  
 final TextEditingController \_amountController = TextEditingController();  
 final TextEditingController \_phoneController = TextEditingController();  
 final TextEditingController \_cnicController = TextEditingController();  
 bool \_isProcessing = false;  
  
 // JazzCash Configuration  
 static const String *merchantId* = "YOUR\_MERCHANT\_ID";  
 static const String *password* = "YOUR\_PASSWORD";  
 static const String *integrityKey* = "YOUR\_INTEGRITY\_SALT";  
 static const String *returnUrl* = "YOUR\_RETURN\_URL";  
  
 String generateSecureHash(Map<String, String> payload) {  
 var sortedKeys = payload.keys.toList()..sort();  
 var hashString = '';  
  
 for (var key in sortedKeys) {  
 if (key != 'pp\_SecureHash' && payload[key]!.isNotEmpty) {  
 hashString += payload[key]! + '&';  
 }  
 }  
  
 hashString = hashString + *integrityKey*;  
  
 var bytes = utf8.encode(hashString);  
 var hash = sha256.convert(bytes);  
  
 return hash.toString().toUpperCase();  
 }  
  
 Future<void> processJazzCashPayment(String amount, String phone, String cnic) async {  
 setState(() {  
 \_isProcessing = true;  
 });  
  
 try {  
 final txnDateTime = DateTime.now();  
 final formattedDate = "${txnDateTime.year}${txnDateTime.month.toString().padLeft(2, '0')}${txnDateTime.day.toString().padLeft(2, '0')}${txnDateTime.hour.toString().padLeft(2, '0')}${txnDateTime.minute.toString().padLeft(2, '0')}${txnDateTime.second.toString().padLeft(2, '0')}";  
 final txnRefNo = "T${DateTime.now().millisecondsSinceEpoch}";  
  
 final payload = {  
 "pp\_Version": "1.1",  
 "pp\_TxnType": "MWALLET",  
 "pp\_Language": "EN",  
 "pp\_MerchantID": *merchantId*,  
 "pp\_SubMerchantID": "",  
 "pp\_Password": *password*,  
 "pp\_BankID": "",  
 "pp\_ProductID": "",  
 "pp\_TxnRefNo": txnRefNo,  
 "pp\_Amount": (double.*parse*(amount) \* 100).toStringAsFixed(0),  
 "pp\_TxnCurrency": "PKR",  
 "pp\_TxnDateTime": formattedDate,  
 "pp\_BillReference": "billRef",  
 "pp\_Description": "Payment for Order",  
 "pp\_MobileNumber": phone,  
 "pp\_CNIC": cnic,  
 "pp\_ReturnURL": *returnUrl*,  
 };  
  
 payload["pp\_SecureHash"] = generateSecureHash(payload);  
//jazzcash error handling  
 final url = Uri.*parse*('https://sandbox.jazzcash.com.pk/CustomerPortal/transactionmanagement/merchantform/');  
  
 final response = await http.post(  
 url,  
 body: payload,  
 headers: {"Content-Type": "application/x-www-form-urlencoded"},  
 );  
  
 if (response.statusCode == 200) {  
 showDialog(  
 context: context,  
 builder: (context) => AlertDialog(  
 title: const Text("Payment Initiated"),  
 content: const Text("Please complete the payment in the JazzCash app"),  
 actions: [  
 TextButton(  
 onPressed: () => Navigator.*pop*(context),  
 child: const Text("OK", style: TextStyle(color: Colors.*black*)),  
 ),  
 ],  
 ),  
 );  
 } else {  
 throw Exception('Payment failed with status: ${response.statusCode}');  
 }  
 } catch (e) {  
 ScaffoldMessenger.*of*(context).showSnackBar(  
 SnackBar(content: Text("Error: $e"), backgroundColor: Colors.*black*),  
 );  
 } finally {  
 setState(() {  
 \_isProcessing = false;  
 });  
 }  
 }  
  
 @override  
 Widget build(BuildContext context) {  
 final conference = widget.conference;  
  
 return Scaffold(  
 backgroundColor: Colors.*white*,  
 appBar: AppBar(  
 title: Text("Pay for ${conference['title']}"),  
 backgroundColor: Colors.*black*,  
 foregroundColor: Colors.*white*,  
 ),  
 body: SingleChildScrollView(  
 child: Padding(  
 padding: const EdgeInsets.all(16.0),  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.start,  
 children: [  
 Text(  
 "Register for ${conference['title']}",  
 style: const TextStyle(  
 fontSize: 22,  
 fontWeight: FontWeight.*bold*,  
 color: Colors.*black*,  
 ),  
 ),  
 const SizedBox(height: 10),  
 Text(  
 "Fees: PKR ${conference['fees']}",  
 style: const TextStyle(fontSize: 18, color: Colors.*black*),  
 ),  
 const SizedBox(height: 20),  
 TextFormField(  
 controller: \_amountController,  
 decoration: const InputDecoration(  
 labelText: "Amount (PKR)",  
 hintText: "Enter amount to pay",  
 border: OutlineInputBorder(),  
 focusedBorder: OutlineInputBorder(  
 borderSide: BorderSide(color: Colors.*black*),  
 ),  
 labelStyle: TextStyle(color: Colors.*black*),  
 ),  
 keyboardType: TextInputType.*number*,  
 ),  
 const SizedBox(height: 16),  
 TextFormField(  
 controller: \_phoneController,  
 decoration: const InputDecoration(  
 labelText: "Phone Number",  
 hintText: "Enter JazzCash account number",  
 border: OutlineInputBorder(),  
 focusedBorder: OutlineInputBorder(  
 borderSide: BorderSide(color: Colors.*black*),  
 ),  
 labelStyle: TextStyle(color: Colors.*black*),  
 ),  
 keyboardType: TextInputType.*phone*,  
 ),  
 const SizedBox(height: 16),  
 TextFormField(  
 controller: \_cnicController,  
 decoration: const InputDecoration(  
 labelText: "CNIC",  
 hintText: "Enter CNIC without dashes",  
 border: OutlineInputBorder(),  
 focusedBorder: OutlineInputBorder(  
 borderSide: BorderSide(color: Colors.*black*),  
 ),  
 labelStyle: TextStyle(color: Colors.*black*),  
 ),  
 keyboardType: TextInputType.*number*,  
 ),  
 const SizedBox(height: 24),  
 \_isProcessing  
 ? const Center(  
 child: CircularProgressIndicator(  
 color: Colors.*black*,  
 ),  
 )  
 : SizedBox(  
 width: double.*infinity*,  
 child: ElevatedButton(  
 onPressed: () {  
 final amount = \_amountController.text.trim();  
 final phone = \_phoneController.text.trim();  
 final cnic = \_cnicController.text.trim();  
  
 if (amount.isEmpty || phone.isEmpty || cnic.isEmpty) {  
 ScaffoldMessenger.*of*(context).showSnackBar(  
 const SnackBar(  
 content: Text("Please fill all fields"),  
 backgroundColor: Colors.*white*,  
 ),  
 );  
 return;  
 }  
  
 processJazzCashPayment(amount, phone, cnic);  
 },  
 style: ElevatedButton.*styleFrom*(  
 backgroundColor: Colors.*black*,  
 foregroundColor: Colors.*white*,  
 padding: const EdgeInsets.symmetric(vertical: 16.0),  
 shape: RoundedRectangleBorder(  
 borderRadius: BorderRadius.circular(8),  
 ),  
 ),  
 child: const Text(  
 "Pay with JazzCash",  
 style: TextStyle(fontSize: 18),  
 ),  
 ),  
 ),  
 ],  
 ),  
 ),  
 ),  
 );  
 }  
}

//MAIN DART

import 'package:flutter/material.dart';  
import 'chat\_screen.dart';  
import 'conferences\_screen.dart';  
  
import 'package:firebase\_core/firebase\_core.dart';  
import 'firebase\_options.dart';  
import 'splash\_screen.dart';  
  
void main() async {  
 WidgetsFlutterBinding.*ensureInitialized*();  
 await Firebase.*initializeApp*(  
 options: DefaultFirebaseOptions.*currentPlatform*,  
 );// Initialize Firebase  
 runApp(const MyApp());  
}  
  
class MyApp extends StatelessWidget {  
 const MyApp({super.key});  
  
 @override  
 Widget build(BuildContext context) {  
 return MaterialApp(  
 debugShowCheckedModeBanner: false,  
 title: 'Flutter Chatbot',  
 theme: ThemeData(  
 brightness: Brightness.light,  
 primarySwatch: Colors.*teal*,  
 ),  
 darkTheme: ThemeData(  
 brightness: Brightness.dark,  
 primarySwatch: Colors.*teal*,  
 ),  
 themeMode: ThemeMode.system,  
 initialRoute: '/',  
 routes: {  
 // '/login': (context) => LoginScreen(),  
 // '/signup': (context) => SignupScreen(),  
 // '/forgot-password': (context) => ForgotPasswordScreen(),  
 '/': (context) => SplashScreen(),  
 '/conferences': (context) => ConferenceScreen(),  
 },  
 );  
 }  
}

// CHEAT SHEET DART

import 'package:flutter/material.dart';  
import 'package:font\_awesome\_flutter/font\_awesome\_flutter.dart';  
class CheatSheet extends StatefulWidget {  
 const CheatSheet({super.key});  
  
 @override  
 State<CheatSheet> createState() => \_CheatSheetState();  
}  
  
class \_CheatSheetState extends State<CheatSheet> {  
  
 @override  
 Widget build(BuildContext context) {  
 return Scaffold(  
 appBar: AppBar(  
 backgroundColor: Colors.*black*,  
 centerTitle: true,  
 elevation: 0,  
 title: const Text('MyMun Cheat Sheet',style: TextStyle(color: Colors.*white*,  
 fontWeight: FontWeight.*bold*,  
 letterSpacing: 1,  
 ),),  
 ),  
 backgroundColor: Colors.*white*,  
 body: Container(  
 height: MediaQuery.*of*(context).size.height,  
 width: MediaQuery.*of*(context).size.width,  
 child: SingleChildScrollView(  
 child: Column(  
 children: [  
 const SizedBox(height: 10,),  
 Container(  
 height: 150,  
 width: 150,  
 decoration: const BoxDecoration(  
 color: Colors.*white*,  
 image: DecorationImage(image: NetworkImage('https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcSLuMGU20iqUxRuhk7DJRuT2T9J0H4iXZ3W4Q&s'),)  
 ),  
 ),  
 const SizedBox(height: 5,),  
 const Center(  
 child: Text('Model UN Cheat Sheet',style: TextStyle(color: Colors.*black87*,fontWeight: FontWeight.*bold*,fontSize: 43),),  
 ),  
 const SizedBox(height: 5,),  
 const Text("You're new to MUN or just quickly need to look up one\n\t of the many specialized terms? This is your place.",style: TextStyle(color: Colors.*black87*,fontSize: 18,fontWeight: FontWeight.*w400*,),)  
 ,const SizedBox(height: 10,),  
 Padding(  
 padding: const EdgeInsets.all(10.0),  
 child: Container(  
 padding: const EdgeInsets.symmetric(horizontal: 10),  
 decoration: BoxDecoration(  
 color: Colors.*grey*[200],  
 borderRadius: BorderRadius.circular(20),  
 border: Border.all(color: Colors.*grey*),  
 ),  
 child: const Row(  
 children: [  
 Icon(Icons.*search*, color: Colors.*grey*),  
 SizedBox(width: 8),  
 Expanded(  
 child: TextField(  
 decoration: InputDecoration(  
 border: InputBorder.*none*,  
 hintText: 'Which word are you looking for?',  
 hintStyle: TextStyle(color: Colors.*grey*),  
 ),  
 ),  
 ),  
 ],  
 ),  
 ),  
 ),  
 const Row(  
 children: [  
 SizedBox(width: 16,),  
 Text('A',style: TextStyle(color: Colors.*black*,fontSize: 24,fontWeight: FontWeight.*bold*),),  
 ],  
 ),  
 const SizedBox(height: 5,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Abstain from Voting / Abstention',style: TextStyle(color: Colors.*black87*,fontWeight: FontWeight.*w800*,fontSize: 18),),  
 ]  
 ),  
 const SizedBox(height: 5,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('When a country abstains from voting in a substantive vote, it\nmeans that the vote is effectively not counted. The country\nwithdraws from having to take a position in favour or against the\nmatter up for voting. It is typically not possible to abstain during\nprocedural votes.',style: TextStyle(color: Colors.*black*,fontWeight: FontWeight.*w400*,fontSize: 16),)  
 ],  
 ),  
 const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Agenda',style: TextStyle(color: Colors.*black*,fontWeight: FontWeight.*bold*,fontSize: 21),),  
 ],  
 ),  
 const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('The topic which is to be discussed in committee. It is typically set\nthrough a "Motion to set the agenda".',style: TextStyle(color: Colors.*black*,fontSize: 16),)  
 ],  
 ),  
 const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Amendment',style: TextStyle(color: Colors.*black*,fontSize: 21,fontWeight: FontWeight.*bold*),)  
 ],  
 ),  
 const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('A change to an already introduced draft resolution. It may\nremove, add or change existing operative clauses.',style: TextStyle(color: Colors.*black*,fontSize: 16,),)  
 ],  
 ),  
 const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Awards',style: TextStyle(color: Colors.*black*,fontSize: 21,fontWeight: FontWeight.*bold*),)  
 ],  
 ),  
 const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('At most conferences, the best delegates of a committee are\nrecognized for their performance. Typically the chairs pick the\nbest delegate of their committee and a few runner-ups.',style: TextStyle(color: Colors.*black*,fontSize: 16,),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('B',style: TextStyle(color: Colors.*black*,fontSize: 24,fontWeight: FontWeight.*bold*),)  
 ],  
 ),  
 const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Bloc',style: TextStyle(color: Colors.*black*,fontSize: 19,fontWeight: FontWeight.*bold*),)  
 ],  
 ),  
 const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('A group of countries that share similar interests or positions.\nThese countries often work together during the conference to\ncreate and pass resolutions.',style: TextStyle(color: Colors.*black*,fontSize: 16),)  
 ],  
 ),  
 const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('C',style: TextStyle(color: Colors.*black*,fontSize: 24,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Caucus',style: TextStyle(color: Colors.*black*,fontSize: 19,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text("A meeting of delegates representing specific countries, often\nheld to discuss specific issues or strategies. There are two types:\nmoderated (structured with a speaker's list) and unmoderated\n(informal, free debate)",style: TextStyle(color: Colors.*black*,fontSize: 16),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Chair',style: TextStyle(color: Colors.*black*,fontSize: 19,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Not a piece of furniture. Impartial moderator of the debate and\nmentor to delegates. Typically a committee has multiple chairs to\nhandle all required tasks.',style: TextStyle(color: Colors.*black*,fontSize: 16),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Committee',style: TextStyle(color: Colors.*black*,fontSize: 19,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('One of the usually many simulated bodies of the United Nations\nwhich focuses on a specific topic in international relations.\nCommittees can be as small as having only 15 people, like the\nSecurity Council or as large as 400, like a double-delegate\nGeneral Assembly.',style: TextStyle(color: Colors.*black*,fontSize: 16),)  
 ],  
 ),  
 const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Consensus',style: TextStyle(color: Colors.*black*,fontSize: 19,fontWeight: FontWeight.*bold*),)  
 ],  
 ),  
 const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Agreement among all parties. In some committees, a resolution\nmay be passed by consensus rather than by a formal vote.',style: TextStyle(color: Colors.*black*,fontSize: 16),)  
 ],  
 ),  
 const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Credentials Committee',style: TextStyle(color: Colors.*black*,fontSize: 19,fontWeight: FontWeight.*bold*),)  
 ],  
 ),  
 const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('A committee that deals with disputes over the rights and\nprivileges of member states. It can also be called upon to resolve\nissues such as the improper use of placards.',style: TextStyle(color: Colors.*black*,fontSize: 16),)  
 ],  
 ),  
 const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('D',style: TextStyle(color: Colors.*black*,fontSize: 24,fontWeight: FontWeight.*bold*),)  
 ],  
 ),  
 const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Dais',style: TextStyle(color: Colors.*black*,fontSize: 19,fontWeight: FontWeight.*bold*),)  
 ],  
 ),  
 const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('The group of staff members who manage the committee, usually\nincluding a chair, a co-chair, and a rapporteur.',style: TextStyle(color: Colors.*black*,fontSize: 16),)  
 ],  
 ),  
 const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Decorum',style: TextStyle(color: Colors.*black*,fontSize: 19,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('The order and respect for others that all delegates are expected\nto show during formal sessions. Chairs often use the phrase\n"Decorum, delegates!" to request delegates to be silent.',style: TextStyle(color: Colors.*black*,fontSize: 16),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Delegate',style: TextStyle(color: Colors.*black*,fontSize: 19,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('A participant in an MUN conference who represents a particular\nmember state, non-governmental organization, or political figure\nin a simulated committee.',style: TextStyle(color: Colors.*black*,fontSize: 16),)  
 ],  
 ),  
 const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Diplomatic Courtesy',style: TextStyle(color: Colors.*black*,fontSize: 19,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Delegates are expected to act respectfully and professionally\nthroughout the conference, as they are representing their\nrespective countries. This is referred to as diplomatic courtesy.',style: TextStyle(color: Colors.*black*,fontSize: 16),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Division of the Question',style: TextStyle(color: Colors.*black*,fontSize: 19,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('During voting on a draft resolution, delegates may move to vote\non different parts or clauses of a resolution separately. This is\ncalled division of the question.',style: TextStyle(color: Colors.*black*,fontSize: 16),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Draft Resolution',style: TextStyle(color: Colors.*black*,fontSize: 19,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text("Already fulfilling all formal resolution requirements, a draft\nresolution remains a draftuntil it is voted upon after the closure\nof the debate. If it is adopted, it becomes an actual resolution,\nand the 'draft' prefix is dropped.",style: TextStyle(color: Colors.*black*,fontSize: 16),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('F',style: TextStyle(color: Colors.*black*,fontSize: 24,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Flow of Debate',style: TextStyle(color: Colors.*black*,fontSize: 19,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('The order of events in a Model UN committee, generally: roll call,\nsetting the agenda, formal debate, presentation of draft\nresolutions, amendments, voting, and adjournment.',style: TextStyle(color: Colors.*black*,fontSize: 16),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Formal Debate',style: TextStyle(color: Colors.*black*,fontSize: 19,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text("The standard format of Model UN conferences, where delegates\nspeak for a set time following a set speaker's list.",style: TextStyle(color: Colors.*black*,fontSize: 16),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Friendly Amendment',style: TextStyle(color: Colors.*black*,fontSize: 19,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('An amendment to a draft resolution that all sponsors agree with.\nIt can be incorporated into the resolution without a vote.',style: TextStyle(color: Colors.*black*,fontSize: 16,),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('G',style: TextStyle(color: Colors.*black*,fontSize: 24,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Gavel',style: TextStyle(color: Colors.*black*,fontSize: 19,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text("A small hammer used by the chairperson to call for attention or\nsignal the beginning or end of a session. It's also a common\naward given to the best delegate in a committee.",style: TextStyle(color: Colors.*black*,fontSize: 16),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text("General Speaker's List",style: TextStyle(color: Colors.*black*,fontSize: 19,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('A list of countries which will speak in order one after the other.\nThe GSL persists for as long as the agenda item is debated.\nThough it can be interrupted by different modes of debate, such\nas moderated or unmoderated caucuses, it continues to persist\nin the background. It will be returned once the dedicated mode\nof debate has elapsed.',style: TextStyle(color: Colors.*black*,fontSize: 16),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('I',style: TextStyle(color: Colors.*black*,fontSize: 24,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('In favour',style: TextStyle(color: Colors.*black*,fontSize: 19,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('When you are for something, not against it, you are in favour of\nit.',style: TextStyle(color: Colors.*black*,fontSize: 16),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Informal Debate',style: TextStyle(color: Colors.*black*,fontSize: 19,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Less structured than formal debate, often used in unmoderated\ncaucuses to allow free-flowing discussion.',style: TextStyle(color: Colors.*black*,fontSize: 16),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('M',style: TextStyle(color: Colors.*black*,fontSize: 14,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Moderated Caucus',style: TextStyle(color: Colors.*black*,fontSize: 19,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text("A moderated caucus is an intense debate lasting for a specified\namount of time (usually between 5 and 15 minutes) to discuss a\nspecific aspect of the overall agenda. Speeches are shorter than\nin the general speaker's list.",style: TextStyle(color: Colors.*black*,fontSize: 16),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Motion',style: TextStyle(color: Colors.*black*,fontSize: 19,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('A suggestion to change something about current committee\nprocedures which is brought forward by one of the delegates: if\nthe motion is granted by the chairs and a majority of delegates,\n the change will be adopted.',style: TextStyle(color: Colors.*black*,fontSize: 16),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('N',style: TextStyle(color: Colors.*black*,fontSize: 24,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Note Passing',style: TextStyle(color: Colors.*black*,fontSize: 19,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('A method used by delegates to communicate with each other\nduring formal debate. This is typically facilitated by the\nconference staff.',style: TextStyle(color: Colors.*black*,fontSize: 16),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('O',style: TextStyle(color: Colors.*black*,fontSize: 24,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Observer',style: TextStyle(color: Colors.*black*,fontSize: 19,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('A participant that represents a non-member state or\norganization, such as the Red Cross or the Vatican. Observers can\nparticipate in debates but usually do not have voting rights.',style: TextStyle(color: Colors.*black*,fontSize: 16),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Operative Clause',style: TextStyle(color: Colors.*black*,fontSize: 19,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('The part of a resolution that describes the actions or policies\nthat the committee will adopt to resolve the issue.',style: TextStyle(color: Colors.*black*,fontSize: 16),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('P',style: TextStyle(color: Colors.*black*,fontSize: 24,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Placard (not Playcard)',style: TextStyle(color: Colors.*black*,fontSize: 19,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text("A little sign stating your country's name. In committee sessions,\nit is used, so chairs know which country you represent. You can\nraise it to indicate to the chairs that you want to raise a point or\na motion.",style: TextStyle(color: Colors.*black*,fontSize: 16,),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Point of Information',style: TextStyle(color: Colors.*black*,fontSize: 19,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text("This really is just another word for 'question'. Points of\ninformation may be allowed after speeches on the General\nSpeaker's list or at some conferences also directed at the chair.",style: TextStyle(color: Colors.*black*,fontSize: 16),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Point of No Return',style: TextStyle(color: Colors.*black*,fontSize: 19,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('A point in the debate after which no more points or motions are\nin order, usually after a motion to close the debate has been\npassed',style: TextStyle(color: Colors.*black*,fontSize: 16),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Point of Order',style: TextStyle(color: Colors.*black*,fontSize: 19,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text("When the chair has made an error, or there was some other error\nin the committee's proceedings, this can be raised to the chair\nrequesting a correction of the error.",style: TextStyle(color: Colors.*black*,fontSize: 16,),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Point of Parliamentary Inquiry',style: TextStyle(color: Colors.*black*,fontSize: 19,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('This is a request to the chair to clarify the rules of procedure.',style: TextStyle(color: Colors.*black*,fontSize: 16,fontWeight: FontWeight.*w100*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Point of Personal Privilege',style: TextStyle(color: Colors.*black*,fontSize: 19,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text("When raised to the chair, this point allows delegates to\ncommunicate concerns regarding their personal well-being (too\nwarm, too cold, can't hear, can't read...) to the chair in hopes of\nimprovement of the situation.",style: TextStyle(color: Colors.*black*,fontSize: 16,),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Position Paper',style: TextStyle(color: Colors.*black*,fontSize: 19,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text("A document prepared before the conference outlining a\ncountry's position on the agenda item or items. It ideally states\nhow the country feels about the current state of affairs and also\nhints at possible solutions it envisions.",style: TextStyle(color: Colors.*black*,fontSize: 16,),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Preambulatory Clause',style: TextStyle(color: Colors.*black*,fontSize: 19,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('The part of a resolution that describes the problem being\naddressed, the past international actions, and the purpose of the\nresolution.',style: TextStyle(color: Colors.*black*,fontSize: 16,),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Precedence',style: TextStyle(color: Colors.*black*,fontSize: 19,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('The order in which motions or points will be considered by the\nchair, usually based on their importance or disruption to the flow\nof debate.',style: TextStyle(color: Colors.*black*,fontSize: 16),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Q',style: TextStyle(color: Colors.*black*,fontSize: 24,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Quorum',style: TextStyle(color: Colors.*black*,fontSize: 19,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text("The quorum is the amount of countries present which is required\nfor the committee to conduct its business. Typically it is a certain\nshare of the countries present during the conference's first roll\ncall.",style: TextStyle(color: Colors.*black*,fontSize: 16),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('R',style: TextStyle(color: Colors.*black*,fontSize: 19,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Rapporteur',style: TextStyle(color: Colors.*black*,fontSize: 19,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text("A member of the dais whose role is to assist in the administrative\ntasks of the committee, such as maintaining the speakers' list or\npassing notes.",style: TextStyle(color: Colors.*black*,fontSize: 16,),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Resolution',style: TextStyle(color: Colors.*black*,fontSize: 19,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text("The document detailing the measures your committee has\nagreed upon to tackle the committee's agenda. Resolutions are\nwritten as one long continuous sentence and must fulfil a\nspecific format. They generally contain two sections: the\npreambulatory clauses to justify the action taken in the\nresolution and the operative clauses which take action.",style: TextStyle(color: Colors.*black*,fontSize: 16,),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Right of Reply',style: TextStyle(color: Colors.*black*,fontSize: 19,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text("A right given to a delegate to respond to serious insults against\ntheir country. It is granted at the discretion of the chair.",style: TextStyle(color: Colors.*black*,fontSize: 16,),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Roll Call',style: TextStyle(color: Colors.*black*,fontSize: 19,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text("The process of checking who is there and who isn't. During a roll\ncall, the chair will check for each country which should be part of\nthe committee, whether it is absent (not there), present (there\nand participating fully), or present and voting (there and\ncommitting not only to vote infavor or against at the end).",style: TextStyle(color: Colors.*black*,fontSize: 16),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Rules of Procedure',style: TextStyle(color: Colors.*black*,fontSize: 19,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text("A document outlining how the specific conference works and\nwhat guidelines have to be followed during the debate. They\ntypically vary to different degrees between conferences, so it's\nworth taking a look every time you go to a new conference!",style: TextStyle(color: Colors.*black*,fontSize: 16),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('S',style: TextStyle(color: Colors.*black*,fontSize: 24,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Secretary General',style: TextStyle(color: Colors.*black*,fontSize: 19,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text("While in the real world, a Secretary General is the head of the\nUN, in Model UN, he or she is usually the chief organizer or head\nof the conference's academics.",style: TextStyle(color: Colors.*black*,fontSize: 16),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Signatories',style: TextStyle(color: Colors.*black*,fontSize: 19,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Countries that support putting a draft resolution on the floor for\ndiscussion. Being a signatory does not necessarily mean the\ndelegate agrees with the resolution, it simply means they would\nlike it to be discussed.',style: TextStyle(color: Colors.*black*,fontSize: 16),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Simple Majority',style: TextStyle(color: Colors.*black*,fontSize: 19,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('A type of voting requirement where a motion passes if more\nmembers vote in favor than against. Abstentions do not count\ntowards the total.',style: TextStyle(color: Colors.*black*,fontSize: 16),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text("Speakers' List",style: TextStyle(color: Colors.*black*,fontSize: 19,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('A list that determines the order of speakers during formal\ndebate. Delegates put their names on this list if they wish to give\na speech.',style: TextStyle(color: Colors.*black*,fontSize: 16),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Sponsors',style: TextStyle(color: Colors.*black*,fontSize: 19,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('The authors of a draft resolution or an amendment. They have\nusually worked together to write the proposal and are the ones\nwho will be advocating for its passage.',style: TextStyle(color: Colors.*black*,fontSize: 16),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Study Guide',style: TextStyle(color: Colors.*black*,fontSize: 19,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('A document provided by the chairs before the conference. It\noffers background info, key issues, and sources to help delegates\nprepare the topic of the committee.',style: TextStyle(color: Colors.*black*,fontSize: 16),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('T',style: TextStyle(color: Colors.*black*,fontSize: 24,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Tabling',style: TextStyle(color: Colors.*black*,fontSize: 19,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text("A motion to suspend discussion on a certain issue or draft\nresolution. The issue can be 'brought off the table' by another\nmotion.",style: TextStyle(color: Colors.*black*,fontSize: 16),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('The Floor',style: TextStyle(color: Colors.*black*,fontSize: 19,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text("The current right to speak given by the chair. A delegate who is\nspeaking is said to 'have the floor'.",style: TextStyle(color: Colors.*black*,fontSize: 16),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('Two-Thirds Majority',style: TextStyle(color: Colors.*black*,fontSize: 19,fontWeight: FontWeight.*bold*),)  
 ],  
 ),const SizedBox(height: 10,),  
 const Row(  
 children: [  
 SizedBox(width: 15,),  
 Text('A type of voting requirement, often used for important\ndecisions. It means that in order for a motion to pass, two-thirds\nof the members must vote in favor.',style: TextStyle(color: Colors.*black*,fontSize: 16),)  
 ],  
 ),  
 SizedBox(height: 10,),  
 Divider(),  
 SizedBox(height: 20,),  
 Container(  
 height: 80,  
 width: MediaQuery.*of*(context).size.width,  
 child: Row(  
 children: [  
 IconButton(  
 onPressed: (){},  
 icon: Icon(FontAwesomeIcons.*tiktok*,color: Colors.*blueGrey*,)  
 ),  
 SizedBox(width: 3,),IconButton(  
 onPressed: (){},  
 icon: Icon(FontAwesomeIcons.*instagram*,color: Colors.*blueGrey*,)  
 ),  
 SizedBox(width: 3,),  
 IconButton(onPressed: (){}, icon: Icon(FontAwesomeIcons.*discord*,color: Colors.*blueGrey*,)),  
 SizedBox(width: 3,),IconButton(  
 onPressed: (){},  
 icon: Icon(FontAwesomeIcons.*linkedin*,color: Colors.*blueGrey*,)  
 ),  
 SizedBox(width: 3,),IconButton(  
 onPressed: (){},  
 icon: Icon(FontAwesomeIcons.*spotify*,color: Colors.*blueGrey*,)  
 ),  
 SizedBox(width: 218,),  
 TextButton(onPressed: (){}, child: Text('Imprint',style: TextStyle(color: Colors.*blueGrey*),))  
 ],  
 ),  
 )  
 ],  
 ),  
 ),  
 ),  
 );  
 }  
}

//LOGIN DART

import 'package:firebase\_auth/firebase\_auth.dart';  
import 'package:flutter/material.dart';  
import 'package:mohsin/authentication/signup.dart';  
import 'package:mohsin/chat\_screen.dart';  
  
class LoginScreen extends StatefulWidget {  
 const LoginScreen({super.key});  
  
 @override  
 State<LoginScreen> createState() => \_LoginScreenState();  
}  
  
class \_LoginScreenState extends State<LoginScreen> {  
 TextEditingController email = TextEditingController();  
 TextEditingController password = TextEditingController();  
  
 @override  
 Widget build(BuildContext context) {  
 return Container(  
 decoration: const BoxDecoration(  
 image: DecorationImage(  
 image: AssetImage('assets/images/login.png'),  
 fit: BoxFit.cover,  
 ),  
 ),  
 child: Scaffold(  
 backgroundColor: Colors.*transparent*,  
 body: Stack(  
 children: [  
 Container(  
 padding: const EdgeInsets.only(left: 35, top: 120),  
 child: const Text(  
 'Welcome\nBack',  
 style: TextStyle(  
 color: Colors.*white*,  
 fontSize: 33,  
 fontWeight: FontWeight.*bold*,  
 ),  
 ),  
 ),  
 SingleChildScrollView(  
 child: Container(  
 padding: EdgeInsets.only(  
 top: MediaQuery.*of*(context).size.height \* 0.5),  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.start,  
 children: [  
 Container(  
 margin: const EdgeInsets.symmetric(horizontal: 35),  
 child: Column(  
 children: [  
 TextField(  
 controller: email,  
 style: const TextStyle(color: Colors.*black*),  
 decoration: InputDecoration(  
 filled: true,  
 fillColor: Colors.*white*,  
 hintText: "Enter Email",  
 hintStyle: TextStyle(color: Colors.*grey*.shade500),  
 labelText: "Email",  
 labelStyle: const TextStyle(color: Colors.*black*),  
 enabledBorder: OutlineInputBorder(  
 borderSide: const BorderSide(color: Colors.*black*),  
 borderRadius: BorderRadius.circular(15),  
 ),  
 focusedBorder: OutlineInputBorder(  
 borderSide: const BorderSide(color: Colors.*black*, width: 2.0),  
 borderRadius: BorderRadius.circular(15),  
 ),  
 ),  
 ),  
 const SizedBox(height: 20),  
 TextField(  
 controller: password,  
 obscureText: true,  
 style: const TextStyle(color: Colors.*black*),  
 decoration: InputDecoration(  
 filled: true,  
 fillColor: Colors.*white*,  
 hintText: "Password",  
 hintStyle: TextStyle(color: Colors.*grey*.shade500),  
 labelText: "Password",  
 labelStyle: const TextStyle(color: Colors.*black*),  
 enabledBorder: OutlineInputBorder(  
 borderSide: const BorderSide(color: Colors.*black*),  
 borderRadius: BorderRadius.circular(15),  
 ),  
 focusedBorder: OutlineInputBorder(  
 borderSide: const BorderSide(color: Colors.*black*, width: 2.0),  
 borderRadius: BorderRadius.circular(15),  
 ),  
 ),  
 ),  
 const SizedBox(height: 40),  
 Row(  
 mainAxisAlignment: MainAxisAlignment.spaceBetween,  
 children: [  
 const Text(  
 'Sign in',  
 style: TextStyle(  
 color: Colors.*black*,  
 fontSize: 27,  
 fontWeight: FontWeight.*w700*,  
 ),  
 ),  
 CircleAvatar(  
 radius: 30,  
 backgroundColor: const Color(0xff4c505b),  
 child: IconButton(  
 color: Colors.*white*,  
 onPressed: () async {  
 try {  
 await FirebaseAuth.*instance*.signInWithEmailAndPassword(  
 email: email.text,  
 password: password.text,  
 );  
 email.clear();  
 password.clear();  
 Navigator.*push*(  
 context,  
 MaterialPageRoute(  
 builder: (context) => ChatScreen(),  
 ),  
 );  
 } on FirebaseAuthException catch (e) {  
 print(e.message);  
 }  
 },  
 icon: const Icon(Icons.*arrow\_forward*),  
 ),  
 ),  
 ],  
 ),  
 const SizedBox(height: 40),  
 Row(  
 mainAxisAlignment: MainAxisAlignment.spaceBetween,  
 children: [  
 TextButton(  
 onPressed: () {  
 Navigator.*push*(  
 context,  
 MaterialPageRoute(  
 builder: (context) => Signup\_screen(),  
 ),  
 );  
 },  
 child: const Text(  
 'Sign Up',  
 textAlign: TextAlign.left,  
 style: TextStyle(  
 decoration: TextDecoration.*underline*,  
 color: Colors.*black*,  
 fontSize: 18,  
 ),  
 ),  
 ),  
 ],  
 ),  
 ],  
 ),  
 ),  
 ],  
 ),  
 ),  
 ),  
 ],  
 ),  
 ),  
 );  
 }  
}

//SIN-UP DART

import 'package:cloud\_firestore/cloud\_firestore.dart';  
import 'package:firebase\_auth/firebase\_auth.dart';  
import 'package:flutter/material.dart';  
import 'package:mohsin/authentication/login.dart';  
  
  
class Signup\_screen extends StatefulWidget {  
 const Signup\_screen({super.key});  
  
 @override  
 State<Signup\_screen> createState() => \_Signup\_screenState();  
}  
  
class \_Signup\_screenState extends State<Signup\_screen> {  
 TextEditingController email = TextEditingController();  
 TextEditingController password = TextEditingController();  
 @override  
 Widget build(BuildContext context) {  
 return Container(  
 decoration: BoxDecoration(  
 image: DecorationImage(  
 image: AssetImage('assets/images/signup.png'), fit: BoxFit.cover),  
 ),  
 child: Scaffold(  
 backgroundColor: Colors.*transparent*,  
 appBar: AppBar(  
 automaticallyImplyLeading: false,  
 backgroundColor: Colors.*transparent*,  
 elevation: 0,  
 ),  
 body: Stack(  
 children: [  
 Container(  
 padding: EdgeInsets.only(left: 35, top: 30),  
 child: Text(  
 'Create\nAccount',  
 style: TextStyle(color: Colors.*white*, fontSize: 33),  
 ),  
 ),  
 SingleChildScrollView(  
 child: Container(  
 padding: EdgeInsets.only(  
 top: MediaQuery.*of*(context).size.height \* 0.28),  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.start,  
 children: [  
 Container(  
 margin: EdgeInsets.only(left: 35, right: 35),  
 child: Column(  
 children: [  
 TextField(  
 controller: email,  
 style: TextStyle(color: Colors.*black*),  
 decoration: InputDecoration(  
 enabledBorder: OutlineInputBorder(  
 borderRadius: BorderRadius.circular(10),  
 borderSide: BorderSide(  
 color: Colors.*black*,  
 ),  
 ),  
 focusedBorder: OutlineInputBorder(  
 borderRadius: BorderRadius.circular(10),  
 borderSide: BorderSide(  
 color: Colors.*black*,  
 ),  
 ),  
 hintText: "Email",  
 hintStyle: TextStyle(color: Colors.*black*),  
 border: OutlineInputBorder(  
 borderRadius: BorderRadius.circular(10),  
 )),  
 ),  
 SizedBox(  
 height: 30,  
 ),  
 TextField(  
 controller: password,  
 style: TextStyle(color: Colors.*black*),  
 obscureText: true,  
 decoration: InputDecoration(  
 enabledBorder: OutlineInputBorder(  
 borderRadius: BorderRadius.circular(10),  
 borderSide: BorderSide(  
 color: Colors.*black*,  
 ),  
 ),  
 focusedBorder: OutlineInputBorder(  
 borderRadius: BorderRadius.circular(10),  
 borderSide: BorderSide(  
 color: Colors.*black*,  
 ),  
 ),  
 hintText: "Password",  
 hintStyle: TextStyle(color: Colors.*black*),  
 border: OutlineInputBorder(  
 borderRadius: BorderRadius.circular(10),  
 )),  
 ),  
 SizedBox(  
 height: 40,  
 ),  
 Row(  
 mainAxisAlignment: MainAxisAlignment.spaceBetween,  
 children: [  
 Text(  
 'Sign Up',  
 style: TextStyle(  
 color: Colors.*white*,  
 fontSize: 27,  
 fontWeight: FontWeight.*w700*),  
 ),  
 CircleAvatar(  
 radius: 30,  
 backgroundColor: Color(0xff4c505b),  
 child: IconButton(  
 color: Colors.*white*,  
 onPressed: () async {  
 try {  
 await FirebaseAuth.*instance* .createUserWithEmailAndPassword(  
 email: email.text,  
 password: password.text);  
 CollectionReference alluser =  
 FirebaseFirestore.*instance* .collection("All\_User");  
 await alluser  
 .doc("${email.text}")  
 .set({"Email": "${email.text}"});  
 email.clear();  
 password.clear();  
 Navigator.*push*(  
 context,  
 MaterialPageRoute(  
 builder: (context) =>  
 LoginScreen()));  
 } on FirebaseAuthException catch (e) {  
 print(e.message);  
 }  
 },  
 icon: Icon(  
 Icons.*arrow\_forward*,  
 )),  
 )  
 ],  
 ),  
 SizedBox(  
 height: 40,  
 ),  
 Row(  
 mainAxisAlignment: MainAxisAlignment.spaceBetween,  
 children: [  
 TextButton(  
 onPressed: () {  
 Navigator.*push*(  
 context,  
 MaterialPageRoute(  
 builder: (context) =>  
 LoginScreen()));  
 },  
 child: Text(  
 'back to Sign In',  
 textAlign: TextAlign.left,  
 style: TextStyle(  
 decoration: TextDecoration.*underline*,  
 color: Colors.*black*,  
 fontSize: 18),  
 ),  
 style: ButtonStyle(),  
 ),  
 ],  
 )  
 ],  
 ),  
 )  
 ],  
 ),  
 ),  
 ),  
 ],  
 ),  
 ),  
 );  
 }  
}

// REGISTRATION.DART

package io.flutter.plugins;  
  
import androidx.annotation.Keep;  
import androidx.annotation.NonNull;  
import io.flutter.Log;  
  
import io.flutter.embedding.engine.FlutterEngine;  
  
*/\*\*  
 \* Generated file. Do not edit.  
 \* This file is generated by the Flutter tool based on the  
 \* plugins that support the Android platform.  
 \*/*@Keep  
public final class GeneratedPluginRegistrant {  
 private static final String *TAG* = "GeneratedPluginRegistrant";  
 public static void registerWith(@NonNull FlutterEngine flutterEngine) {  
 try {  
 flutterEngine.getPlugins().add(new io.flutter.plugins.firebase.firestore.FlutterFirebaseFirestorePlugin());  
 } catch (Exception e) {  
 Log.e(*TAG*, "Error registering plugin cloud\_firestore, io.flutter.plugins.firebase.firestore.FlutterFirebaseFirestorePlugin", e);  
 }  
 try {  
 flutterEngine.getPlugins().add(new io.flutter.plugins.firebase.auth.FlutterFirebaseAuthPlugin());  
 } catch (Exception e) {  
 Log.e(*TAG*, "Error registering plugin firebase\_auth, io.flutter.plugins.firebase.auth.FlutterFirebaseAuthPlugin", e);  
 }  
 try {  
 flutterEngine.getPlugins().add(new io.flutter.plugins.firebase.core.FlutterFirebaseCorePlugin());  
 } catch (Exception e) {  
 Log.e(*TAG*, "Error registering plugin firebase\_core, io.flutter.plugins.firebase.core.FlutterFirebaseCorePlugin", e);  
 }  
 try {  
 flutterEngine.getPlugins().add(new io.flutter.plugins.pathprovider.PathProviderPlugin());  
 } catch (Exception e) {  
 Log.e(*TAG*, "Error registering plugin path\_provider\_android, io.flutter.plugins.pathprovider.PathProviderPlugin", e);  
 }  
 try {  
 flutterEngine.getPlugins().add(new com.tekartik.sqflite.SqflitePlugin());  
 } catch (Exception e) {  
 Log.e(*TAG*, "Error registering plugin sqflite\_android, com.tekartik.sqflite.SqflitePlugin", e);  
 }  
 try {  
 flutterEngine.getPlugins().add(new io.flutter.plugins.urllauncher.UrlLauncherPlugin());  
 } catch (Exception e) {  
 Log.e(*TAG*, "Error registering plugin url\_launcher\_android, io.flutter.plugins.urllauncher.UrlLauncherPlugin", e);  
 }  
 try {  
 flutterEngine.getPlugins().add(new io.flutter.plugins.videoplayer.VideoPlayerPlugin());  
 } catch (Exception e) {  
 Log.e(*TAG*, "Error registering plugin video\_player\_android, io.flutter.plugins.videoplayer.VideoPlayerPlugin", e);  
 }  
 }  
}

//API SERVICE.DART

import 'dart:convert';  
import 'package:http/http.dart' as http;  
  
class ApiService {  
 static const String *\_baseUrl* = 'https://e64f-34-16-161-120.ngrok-free.app/chat'; // Replace with your backend URL  
  
 static Future<String> *getBotResponse*(String userInput) async {  
 try {  
 final response = await http.post(  
 Uri.*parse*(*\_baseUrl*),  
 headers: {'Content-Type': 'application/json'},  
 body: jsonEncode({'message': userInput}),  
 );  
  
 print('Response status: ${response.statusCode}');  
 print('Response body: ${response.body}');  
  
 if (response.statusCode == 200) {  
 final data = jsonDecode(response.body);  
 return data['response'] ?? 'No response from bot';  
 } else {  
 return 'Error: ${response.reasonPhrase}';  
 }  
 } catch (e) {  
 print('Error: $e');  
 return 'Error: Unable to connect to server';  
 }  
 }  
  
}

//profile dart

import 'package:flutter/material.dart';  
import 'package:firebase\_auth/firebase\_auth.dart';  
import 'package:intl/intl.dart';  
  
import 'authentication/authstream.dart';  
  
class ProfileScreen extends StatelessWidget {  
 final String userEmail;  
 final String userName;  
  
 const ProfileScreen({  
 Key? key,  
 required this.userEmail,  
 required this.userName,  
 }) : super(key: key);  
  
 @override  
 Widget build(BuildContext context) {  
 // Mock data for registered conferences. Replace with your database logic.  
 final List<Map<String, String>> registeredConferences = [  
 {'title': 'Flutter Dev Conference', 'date': '2025-01-20'},  
 {'title': 'AI Summit 2025', 'date': '2025-02-15'},  
 ];  
  
 return Scaffold(  
 backgroundColor: Colors.grey[100],  
 appBar: AppBar(  
 title: const Text('Profile'),  
 backgroundColor: Colors.black,  
 elevation: 0,  
 actions: [  
 IconButton(  
 icon: const Icon(Icons.logout),  
 onPressed: ()  
 {  
 FirebaseAuth \_auth = FirebaseAuth.instance;  
 \_auth.signOut();  
 Navigator.push(context, MaterialPageRoute(builder: (context) => Authstream()));  
 },  
 ),  
 ],  
 ),  
 body: SingleChildScrollView(  
 child: Column(  
 children: [  
 // Profile Header Section  
 Container(  
 decoration: BoxDecoration(  
 color: Colors.black,  
 borderRadius: const BorderRadius.only(  
 bottomLeft: Radius.circular(30),  
 bottomRight: Radius.circular(30),  
 ),  
 ),  
 child: Padding(  
 padding: const EdgeInsets.all(20.0),  
 child: Column(  
 children: [  
 const CircleAvatar(  
 radius: 50,  
 backgroundColor: Colors.white,  
 child: Icon(Icons.person, size: 50, color: Colors.black),  
 ),  
 const SizedBox(height: 16),  
 Text(  
 userName,  
 style: const TextStyle(  
 fontSize: 24,  
 fontWeight: FontWeight.bold,  
 color: Colors.white,  
 ),  
 ),  
 const SizedBox(height: 8),  
 Text(  
 userEmail,  
 style: const TextStyle(  
 fontSize: 16,  
 color: Colors.white70,  
 ),  
 ),  
 const SizedBox(height: 20),  
 ],  
 ),  
 ),  
 ),  
  
 // Conferences Section  
 Padding(  
 padding: const EdgeInsets.all(20.0),  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.start,  
 children: [  
 const Row(  
 mainAxisAlignment: MainAxisAlignment.spaceBetween,  
 children: [  
 Text(  
 'Registered Conferences',  
 style: TextStyle(  
 fontSize: 20,  
 fontWeight: FontWeight.bold,  
 color: Colors.black87,  
 ),  
 ),  
 Chip(  
 label: Text('2 Active'),  
 backgroundColor: Colors.black,  
 labelStyle: TextStyle(color: Colors.white),  
 ),  
 ],  
 ),  
 const SizedBox(height: 16),  
 ...registeredConferences.map((conference) {  
 final formattedDate = DateFormat('MMM dd, yyyy')  
 .format(DateTime.parse(conference['date']!));  
 return Card(  
 elevation: 2,  
 margin: const EdgeInsets.only(bottom: 16),  
 shape: RoundedRectangleBorder(  
 borderRadius: BorderRadius.circular(12),  
 ),  
 child: ListTile(  
 contentPadding: const EdgeInsets.all(16),  
 leading: Container(  
 width: 50,  
 height: 50,  
 decoration: BoxDecoration(  
 color: Colors.black.withOpacity(0.1),  
 borderRadius: BorderRadius.circular(8),  
 ),  
 child: const Icon(  
 Icons.event,  
 color: Colors.black,  
 ),  
 ),  
 title: Text(  
 conference['title']!,  
 style: const TextStyle(  
 fontSize: 18,  
 fontWeight: FontWeight.bold,  
 ),  
 ),  
 subtitle: Column(  
 crossAxisAlignment: CrossAxisAlignment.start,  
 children: [  
 const SizedBox(height: 8),  
 Row(  
 children: [  
 const Icon(Icons.calendar\_today,  
 size: 16,  
 color: Colors.grey),  
 const SizedBox(width: 8),  
 Text(  
 formattedDate,  
 style: const TextStyle(  
 color: Colors.grey,  
 ),  
 ),  
 ],  
 ),  
 ],  
 ),  
 trailing: const Icon(Icons.chevron\_right),  
 onTap: () {  
 // Add navigation to conference details  
 },  
 ),  
 );  
 }).toList(),  
 ],  
 ),  
 ),  
 ],  
 ),  
 ),  
 );  
 }  
}

//