```dart
// Arquivo principal do aplicativo Vibea+.
// Este arquivo contém a inicialização do Firebase e as telas principais do aplicativo.

import 'package:flutter/material.dart';
import 'package:firebase_core/firebase_core.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:google_sign_in/google_sign_in.dart';
import 'package:sign_in_with_apple/sign_in_with_apple.dart';
import 'package:cloud_firestore/cloud_firestore.dart';

// TODO: Certifique-se de adicionar as dependências no seu pubspec.yaml:
// firebase_core: ^2.24.2
// firebase_auth: ^4.15.1
// google_sign_in: ^6.1.6
// sign_in_with_apple: ^4.3.0
// cloud_firestore: ^4.13.6

// O ponto de entrada principal do seu aplicativo Flutter.
void main() async {
  WidgetsFlutterBinding.ensureInitialized();
  await Firebase.initializeApp(); // Inicializa o Firebase.
  runApp(const MyApp());
}

// O widget principal do aplicativo.
class MyApp extends StatelessWidget {
  const MyApp({super.key});

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Vibea+',
      debugShowCheckedModeBanner: false, // Oculta a faixa de debug.
      theme: ThemeData(
        brightness: Brightness.dark,
        scaffoldBackgroundColor: const Color(0xFF000000), // Preto base.
        cardColor: const Color(0xFF0B0B0B),
        colorScheme: const ColorScheme.dark(
          primary: Color(0xFF7C5CFC), // Roxo Vibrante
          onPrimary: Color(0xFFFFFFFF),
          secondary: Color(0xFFB3B3B3),
          surface: Color(0xFF121212),
        ),
        textTheme: const TextTheme(
          bodyLarge: TextStyle(color: Color(0xFFFFFFFF)),
          bodyMedium: TextStyle(color: Color(0xFFB3B3B3)),
          titleLarge: TextStyle(
            color: Color(0xFFFFFFFF),
            fontWeight: FontWeight.bold,
          ),
        ),
        useMaterial3: true,
      ),
      home: AuthScreen(),
    );
  }
}

// A tela de Autenticação inicial do aplicativo.
class AuthScreen extends StatefulWidget {
  const AuthScreen({super.key});
  @override
  State<AuthScreen> createState() => _AuthScreenState();
}

class _AuthScreenState extends State<AuthScreen> {
  final FirebaseAuth _auth = FirebaseAuth.instance;
  final GoogleSignIn _googleSignIn = GoogleSignIn();
  final FirebaseFirestore _firestore = FirebaseFirestore.instance;

  // Função para exibir uma mensagem na parte inferior da tela.
  void _showMessage(String message) {
    ScaffoldMessenger.of(context).showSnackBar(
      SnackBar(
        content: Text(message),
        duration: const Duration(seconds: 2),
      ),
    );
  }

  // Função para salvar os dados do usuário no Firestore após o login.
  Future<void> _saveUserToFirestore(User user) async {
    final userRef = _firestore.collection('users').doc(user.uid);
    final doc = await userRef.get();
    if (!doc.exists) {
      await userRef.set({
        'uid': user.uid,
        'email': user.email,
        'displayName': user.displayName,
        'photoURL': user.photoURL,
        'createdAt': FieldValue.serverTimestamp(),
        'bio': 'Olá, este é o meu perfil no Vibea+!',
        'age': 25,
      });
    }
  }

  // Autenticação com Google
  Future<void> _signInWithGoogle() async {
    try {
      final GoogleSignInAccount? googleUser = await _googleSignIn.signIn();
      if (googleUser == null) {
        _showMessage('Login com Google cancelado.');
        return;
      }
      final GoogleSignInAuthentication googleAuth = await googleUser.authentication;
      final credential = GoogleAuthProvider.credential(
        accessToken: googleAuth.accessToken,
        idToken: googleAuth.idToken,
      );
      final userCredential = await _auth.signInWithCredential(credential);
      if (userCredential.user != null) {
        await _saveUserToFirestore(userCredential.user!);
        _navigateToHome();
      }
    } on FirebaseAuthException catch (e) {
      _showMessage('Erro ao entrar com Google: ${e.message}');
    } catch (e) {
      _showMessage('Erro inesperado: $e');
    }
  }

  // Autenticação com Apple
```

```
Future<void> _signInWithApple() async {
  try {
    final credential = await SignInWithApple.getAppleIDCredential(
      scopes: [
        AppleIDAuthorizationScopes.email,
        AppleIDAuthorizationScopes.fullName,
      ],
    );
    final authCredential = AppleAuthProvider.credential(
      identityToken: credential.identityToken,
      rawNonce: credential.nonce,
    );
    final userCredential = await _auth.signInWithCredential(authCredential);
    if (userCredential.user != null) {
      await _saveUserToFirestore(userCredential.user!);
      _navigateToHome();
    }
  } on FirebaseAuthException catch (e) {
    _showMessage('Erro ao entrar com Apple: ${e.message}');
  } catch (e) {
    _showMessage('Erro inesperado: $e');
  }
}

// Função para navegar para a próxima tela após o login.
void _navigateToHome() {
  Navigator.of(context).pushReplacement(
    MaterialPageRoute(builder: (_) => const HomeScreen()),
  );
}
```

```
@override
Widget build(BuildContext context) {
 return Scaffold(
  body: Center(
   child: Padding(
    padding: const EdgeInsets.symmetric(horizontal: 24.0),
    child: Column(
     mainAxisAlignment: MainAxisAlignment.center,
     children: <Widget>[
      Text(
       'Vibea+',
       style: Theme.of(context).textTheme.titleLarge?.copyWith(fontSize: 48),
      ),
      const SizedBox(height: 16.0),
      const Text(
       'Bem-vindo! Onde seus caminhos se cruzam com as pessoas certas.',
       textAlign: TextAlign.center,
       style: TextStyle(fontSize: 18.0),
      ),
      const SizedBox(height: 48.0),
      Container(
       decoration: BoxDecoration(
        color: Theme.of(context).cardColor,
        borderRadius: BorderRadius.circular(12.0),
        boxShadow: [
         BoxShadow(
          color: Colors.white.withOpacity(0.05),
          offset: const Offset(-4, -4),
          blurRadius: 8,
          spreadRadius: 1,
         ),
    BoxShadow(
      color: Colors.black.withOpacity(0.4),
      offset: const Offset(4, 4),
      blurRadius: 8,
      spreadRadius: 1,
    ),
  ],
),
child: Padding(
  padding: const EdgeInsets.all(24.0),
  child: Column(
    children: [
      Text(
        'Autenticação',
        style: Theme.of(context).textTheme.bodyLarge?.copyWith(
              fontSize: 20,
              fontWeight: FontWeight.bold,
            ),
      ),
      const SizedBox(height: 24.0),
      ElevatedButton.icon(
        onPressed: () {
          _navigateToPhoneAuth();
        },
        style: ElevatedButton.styleFrom(
          foregroundColor: Colors.white,
          backgroundColor: Theme.of(context).colorScheme.primary,
          shape: RoundedRectangleBorder(
            borderRadius: BorderRadius.circular(12),
          ),
          minimumSize: const Size(double.infinity, 50),
  ),
  icon: const Icon(Icons.phone),
  label: const Text(
    'Login por Telefone',
    style: TextStyle(fontSize: 16),
  ),
),
const SizedBox(height: 16.0),
ElevatedButton.icon(
  onPressed: _signInWithGoogle,
  style: ElevatedButton.styleFrom(
    foregroundColor: Colors.black,
    backgroundColor: Colors.white,
    shape: RoundedRectangleBorder(
      borderRadius: BorderRadius.circular(12),
    ),
    minimumSize: const Size(double.infinity, 50),
  ),
  icon: const Icon(Icons.g_mobiledata),
  label: const Text(
    'Entrar com Google',
    style: TextStyle(fontSize: 16),
  ),
),
const SizedBox(height: 16.0),
if (Theme.of(context).platform == TargetPlatform.iOS)
  ElevatedButton.icon(
    onPressed: _signInWithApple,
    style: ElevatedButton.styleFrom(
      foregroundColor: Colors.white,
      backgroundColor: Colors.black,
                            shape: RoundedRectangleBorder(
                              borderRadius: BorderRadius.circular(12),
                            ),
                            minimumSize: const Size(double.infinity, 50),
                          ),
                          icon: const Icon(Icons.apple),
                          label: const Text(
                            'Entrar com Apple',
                            style: TextStyle(fontSize: 16),
                          ),
                        ),
                    ],
                  ),
                ),
              ),
            ],
          ),
        ),
      ),
    );
  }

  // Navega para a tela de autenticação por telefone
  void _navigateToPhoneAuth() {
    Navigator.of(context).push(
      MaterialPageRoute(builder: (_) => const PhoneAuthScreen()),
    );
  }
}

// Uma tela de placeholder para simular a navegação após o login.
```

```
class HomeScreen extends StatefulWidget {
  const HomeScreen({super.key});

  @override
  State<HomeScreen> createState() => _HomeScreenState();
}

class _HomeScreenState extends State<HomeScreen> {
  int _selectedIndex = 0;
  static const List<Widget> _widgetOptions = <Widget>[
    DiscoveryScreen(),
    ChatScreen(),
    ProfileScreen(),
  ];

  void _onItemTapped(int index) {
    setState(() {
      _selectedIndex = index;
    });
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: const Text('Vibea+'),
        backgroundColor: Colors.transparent,
        elevation: 0,
        actions: [
          IconButton(
            icon: const Icon(Icons.logout),
            onPressed: () async {
              await FirebaseAuth.instance.signOut();
              await GoogleSignIn().signOut();
              if (context.mounted) {
                Navigator.of(context).pushReplacement(
                  MaterialPageRoute(builder: (_) => const AuthScreen()),
                );
              }
            },
          ),
        ],
      ),
      body: _widgetOptions.elementAt(_selectedIndex),
      bottomNavigationBar: BottomNavigationBar(
        items: const <BottomNavigationBarItem>[
          BottomNavigationBarItem(
            icon: Icon(Icons.people_alt),
            label: 'Descoberta',
          ),
          BottomNavigationBarItem(
            icon: Icon(Icons.chat),
            label: 'Chat',
          ),
          BottomNavigationBarItem(
            icon: Icon(Icons.account_circle),
            label: 'Perfil',
          ),
        ],
        currentIndex: _selectedIndex,
        onTap: _onItemTapped,
        backgroundColor: Theme.of(context).cardColor,
              selectedItemColor: Theme.of(context).colorScheme.primary,
              unselectedItemColor: Theme.of(context).colorScheme.secondary,
            ),
          );
        }
      }

      // Nova tela para autenticação por telefone.
      class PhoneAuthScreen extends StatefulWidget {
        const PhoneAuthScreen({super.key});

        @override
        State<PhoneAuthScreen> createState() => _PhoneAuthScreenState();
      }

      class _PhoneAuthScreenState extends State<PhoneAuthScreen> {
        final FirebaseAuth _auth = FirebaseAuth.instance;
        final FirebaseFirestore _firestore = FirebaseFirestore.instance;
        final TextEditingController _phoneController = TextEditingController();
        final TextEditingController _otpController = TextEditingController();
        String _verificationId = '';
        bool _otpSent = false;

        void _showMessage(String message) {
          ScaffoldMessenger.of(context).showSnackBar(
            SnackBar(
              content: Text(message),
              duration: const Duration(seconds: 2),
            ),
          );
        }
```

```
Future<void> _verifyPhoneNumber() async {
  try {
    await _auth.verifyPhoneNumber(
      phoneNumber: _phoneController.text,
      verificationCompleted: (PhoneAuthCredential credential) async {
        final userCredential = await _auth.signInWithCredential(credential);
        if (userCredential.user != null) {
          await _saveUserToFirestore(userCredential.user!);
          _navigateToHome();
        }
      },
      verificationFailed: (FirebaseAuthException e) {
        _showMessage('Verificação falhou: ${e.message}');
      },
      codeSent: (String verificationId, int? resendToken) {
        _verificationId = verificationId;
        setState(() {
          _otpSent = true;
        });
        _showMessage('Código de verificação enviado.');
      },
      codeAutoRetrievalTimeout: (String verificationId) {
        _verificationId = verificationId;
      },
      timeout: const Duration(seconds: 60),
    );
  } catch (e) {
    _showMessage('Erro ao verificar o número: $e');
  }
}
```

```
Future<void> _signInWithOtp() async {
 try {
  final credential = PhoneAuthProvider.credential(
   verificationId: _verificationId,
   smsCode: _otpController.text,
  );
  final userCredential = await _auth.signInWithCredential(credential);
  if (userCredential.user != null) {
   await _saveUserToFirestore(userCredential.user!);
   _navigateToHome();
  }
 } on FirebaseAuthException catch (e) {
  _showMessage('Erro ao entrar com o código: ${e.message}');
 }
}

Future<void> _saveUserToFirestore(User user) async {
 final userRef = _firestore.collection('users').doc(user.uid);
 final doc = await userRef.get();
 if (!doc.exists) {
  await userRef.set({
   'uid': user.uid,
   'email': user.email,
   'displayName': 'Usuário ${user.phoneNumber}',
   'photoURL': null,
   'createdAt': FieldValue.serverTimestamp(),
   'bio': 'Olá, este é o meu perfil no Vibea+!',
   'age': 25,
  });
 }
```

```
}

void _navigateToHome() {
  Navigator.of(context).pushReplacement(
    MaterialPageRoute(builder: (_) => const HomeScreen()),
  );
}

@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      title: const Text('Login por Telefone'),
      backgroundColor: Colors.transparent,
      elevation: 0,
    ),
    body: Center(
      child: Padding(
        padding: const EdgeInsets.all(24.0),
        child: Column(
          mainAxisAlignment: MainAxisAlignment.center,
          children: [
            TextField(
              controller: _phoneController,
              keyboardType: TextInputType.phone,
              decoration: InputDecoration(
                labelText: 'Número de Telefone',
                hintText: '+55 11 99999-9999',
                border: OutlineInputBorder(
                  borderRadius: BorderRadius.circular(12),
                ),
  ),
),
const SizedBox(height: 16.0),
ElevatedButton(
  onPressed: _verifyPhoneNumber,
  style: ElevatedButton.styleFrom(
    minimumSize: const Size(double.infinity, 50),
  ),
  child: const Text('Enviar Código de Verificação'),
),
if (_otpSent) ...[
  const SizedBox(height: 24.0),
  TextField(
    controller: _otpController,
    keyboardType: TextInputType.number,
    decoration: InputDecoration(
      labelText: 'Código de Verificação (OTP)',
      border: OutlineInputBorder(
        borderRadius: BorderRadius.circular(12),
      ),
    ),
  ),
  const SizedBox(height: 16.0),
  ElevatedButton(
    onPressed: _signInWithOtp,
    style: ElevatedButton.styleFrom(
      minimumSize: const Size(double.infinity, 50),
      backgroundColor: Theme.of(context).colorScheme.primary,
      foregroundColor: Theme.of(context).colorScheme.onPrimary,
    ),
    child: const Text('Entrar'),
            ),
          ],
        ],
      ),
    ),
  ),
);
}
}

// Nova tela de perfil do usuário.
class ProfileScreen extends StatelessWidget {
 const ProfileScreen({super.key});

 @override
 Widget build(BuildContext context) {
  final user = FirebaseAuth.instance.currentUser;
  if (user == null) {
   return const Center(child: Text('Nenhum usuário logado.'));
  }

  return StreamBuilder<DocumentSnapshot>(
   stream: FirebaseFirestore.instance.collection('users').doc(user.uid).snapshots(),
   builder: (context, snapshot) {
    if (!snapshot.hasData || snapshot.hasError) {
     return const Center(child: CircularProgressIndicator());
    }

    final data = snapshot.data!.data() as Map<String, dynamic>?;
    final name = data?['displayName'] ?? 'Nome do Usuário';
    final email = data?['email'] ?? 'email@example.com';
final photoURL = data?['photoURL'];


return SingleChildScrollView(
  child: Padding(
    padding: const EdgeInsets.all(24.0),
    child: Column(
      crossAxisAlignment: CrossAxisAlignment.center,
      children: [
        CircleAvatar(
          radius: 60,
          backgroundImage: photoURL != null ? NetworkImage(photoURL) : null,
          child: photoURL == null ? const Icon(Icons.person, size: 80) : null,
        ),
        const SizedBox(height: 24),
        Text(
          name,
          style: Theme.of(context).textTheme.titleLarge?.copyWith(fontSize: 28),
          textAlign: TextAlign.center,
        ),
        const SizedBox(height: 8),
        Text(
          email,
          style: Theme.of(context).textTheme.bodyMedium?.copyWith(fontSize: 16),
          textAlign: TextAlign.center,
        ),
        const SizedBox(height: 48),
        ElevatedButton.icon(
          onPressed: () {
            // TODO: Implementar a navegação para a tela de edição de perfil
            // Aqui o usuário poderá atualizar o nome, bio, etc.
          },
                  icon: const Icon(Icons.edit),
                  label: const Text('Editar Perfil'),
                  style: ElevatedButton.styleFrom(
                    minimumSize: const Size(double.infinity, 50),
                    backgroundColor: Theme.of(context).colorScheme.primary,
                    foregroundColor: Theme.of(context).colorScheme.onPrimary,
                    shape: RoundedRectangleBorder(
                      borderRadius: BorderRadius.circular(12),
                    ),
                  ),
                ),
              ],
            ),
          ),
        );
      },
    );
  }
}

// Classe para representar os dados de um usuário para a descoberta.
class UserProfile {
  final String uid;
  final String name;
  final int age;
  final String bio;
  final String imageUrl;

  UserProfile({
    required this.uid,
    required this.name,
    required this.age,
    required this.bio,
    required this.imageUrl,
  });

  factory UserProfile.fromDocument(DocumentSnapshot doc) {
    final data = doc.data() as Map<String, dynamic>;
    return UserProfile(
      uid: doc.id,
      name: data['displayName'] ?? 'Nome Desconhecido',
      age: data['age'] ?? 0,
      bio: data['bio'] ?? 'Nenhuma biografia disponível.',
      imageUrl: data['photoURL'] ??
'https://placehold.co/400x500/0B0B0B/B3B3B3?text=Vibea+',
    );
  }
}

// Nova tela para a funcionalidade de descoberta (swipe).
class DiscoveryScreen extends StatefulWidget {
  const DiscoveryScreen({super.key});

  @override
  State<DiscoveryScreen> createState() => _DiscoveryScreenState();
}

class _DiscoveryScreenState extends State<DiscoveryScreen> {
  final String currentUserId = FirebaseAuth.instance.currentUser!.uid;

  void _showMessage(String message) {
    ScaffoldMessenger.of(context).showSnackBar(
      SnackBar(
        content: Text(message),
        duration: const Duration(seconds: 2),
      ),
    );
  }

  void _onSwipeLeft(String userId) {
    _showMessage('Você descartou o perfil.');
    // TODO: Adicionar lógica para salvar a rejeição no Firestore
  }

  void _onSwipeRight(String userId) async {
    _showMessage('Você curtiu o perfil!');
    // TODO: Adicionar lógica para salvar a curtida no Firestore
    // e verificar se é um "match".
  }

  @override
  Widget build(BuildContext context) {
    return StreamBuilder<QuerySnapshot>(
      stream: FirebaseFirestore.instance.collection('users').snapshots(),
      builder: (context, snapshot) {
        if (snapshot.connectionState == ConnectionState.waiting) {
          return const Center(child: CircularProgressIndicator());
        }
        if (snapshot.hasError) {
          return Center(child: Text('Erro: ${snapshot.error}'));
        }
        if (!snapshot.hasData || snapshot.data!.docs.isEmpty) {
          return const Center(
      child: Text(
        'Não há mais perfis por perto.',
        style: TextStyle(fontSize: 20),
        textAlign: TextAlign.center,
      ),
    );
  }

  final users = snapshot.data!.docs
      .where((doc) => doc.id != currentUserId) // Exclui o próprio usuário
      .map((doc) => UserProfile.fromDocument(doc))
      .toList();

  if (users.isEmpty) {
      return const Center(
      child: Text(
        'Não há mais perfis por perto.',
        style: TextStyle(fontSize: 20),
        textAlign: TextAlign.center,
      ),
    );
  }

  final topUser = users.first;

  return Center(
    child: Padding(
      padding: const EdgeInsets.symmetric(horizontal: 24.0, vertical: 32.0),
      child: Column(
        children: [
          Expanded(
        child: Draggable(
          feedback: UserCard(user: topUser),
          childWhenDragging: Container(),
          onDragEnd: (details) {
            if (details.velocity.pixelsPerSecond.dx < -1000) {
              _onSwipeLeft(topUser.uid);
            } else if (details.velocity.pixelsPerSecond.dx > 1000) {
              _onSwipeRight(topUser.uid);
            }
          },
          child: UserCard(user: topUser),
        ),
      ),
      const SizedBox(height: 24),
      Row(
        mainAxisAlignment: MainAxisAlignment.spaceEvenly,
        children: [
          FloatingActionButton(
            onPressed: () => _onSwipeLeft(topUser.uid),
            backgroundColor: const Color(0xFF1E1E1E),
            child: const Icon(Icons.close, color: Colors.red),
          ),
          FloatingActionButton(
            onPressed: () => _onSwipeRight(topUser.uid),
            backgroundColor: const Color(0xFF1E1E1E),
            child: const Icon(Icons.favorite, color: Colors.green),
          ),
        ],
      ),
    ],
  ),
            ),
          );
        },
      );
    }
  }

  // Widget do cartão de usuário para o modo de descoberta.
  class UserCard extends StatelessWidget {
    final UserProfile user;

    const UserCard({super.key, required this.user});

    @override
    Widget build(BuildContext context) {
      return Card(
        clipBehavior: Clip.antiAlias,
        shape: RoundedRectangleBorder(
          borderRadius: BorderRadius.circular(16.0),
        ),
        child: Stack(
          fit: StackFit.expand,
          children: [
            Image.network(
              user.imageUrl,
              fit: BoxFit.cover,
            ),
            Container(
              decoration: BoxDecoration(
                gradient: LinearGradient(
                  begin: Alignment.topCenter,
        end: Alignment.bottomCenter,
        colors: [
          Colors.transparent,
          Colors.black.withOpacity(0.7),
        ],
      ),
    ),
  ),
  Positioned(
    bottom: 16.0,
    left: 16.0,
    right: 16.0,
    child: Column(
      crossAxisAlignment: CrossAxisAlignment.start,
      children: [
        Text(
          '${user.name}, ${user.age}',
          style: const TextStyle(
            color: Colors.white,
            fontSize: 28,
            fontWeight: FontWeight.bold,
          ),
        ),
        const SizedBox(height: 8),
        Text(
          user.bio,
          style: const TextStyle(
            color: Colors.white,
            fontSize: 16,
          ),
        ),
              ],
            ),
          ),
        ],
      ),
    );
   }
  }

  // Nova tela de Chat.
  class ChatScreen extends StatelessWidget {
    const ChatScreen({super.key});

    @override
    Widget build(BuildContext context) {
      // TODO: Implementar a lógica de chat aqui, usando o Firebase Firestore
      // para exibir as conversas e enviar mensagens.
      return Center(
        child: Column(
          mainAxisAlignment: MainAxisAlignment.center,
          children: [
            const Icon(Icons.chat_bubble_outline, size: 80, color: Color(0xFFB3B3B3)),
            const SizedBox(height: 16),
            Text(
              'Seu chat será exibido aqui!',
              style: Theme.of(context).textTheme.bodyMedium?.copyWith(fontSize: 20),
              textAlign: TextAlign.center,
            ),
          ],
        ),
      );
    }
  }
```