Settings responsive code

```dart
import '/auth/firebase_auth/auth_util.dart';
import '/componenet/app_bar/app_bar_widget.dart';
import '/componenet/side_nav/side_nav_widget.dart';
import '/flutter_flow/flutter_flow_theme.dart';
import '/flutter_flow/flutter_flow_util.dart';
import '/index.dart';
import 'package:flutter/material.dart';
import 'package:font_awesome_flutter/font_awesome_flutter.dart';
import 'settings_model.dart';
export 'settings_model.dart';

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

 @override
 State<SettingsWidget> createState() => _SettingsWidgetState();
}

class _SettingsWidgetState extends State<SettingsWidget> {
 late SettingsModel _model;
 final scaffoldKey = GlobalKey<ScaffoldState>();

 @override
 void initState() {
   super.initState();
   _model = createModel(context, () => SettingsModel());
   _model.switchValue1 = true;
   _model.switchValue2 = true;
   WidgetsBinding.instance.addPostFrameCallback((_) => safeSetState(() {}));
 }

 @override
 void dispose() {
   _model.dispose();
   super.dispose();
 }


 double responsiveFontSize(BuildContext context,
     {double desktop = 22, double tablet = 20, double mobile = 18}) {
   final width = MediaQuery.of(context).size.width;
   if (width > 1200) return desktop;
   if (width > 600) return tablet;
   return mobile;
 }
```

```
EdgeInsets responsivePadding(BuildContext context) {
  final width = MediaQuery.of(context).size.width;
  if (width > 1200) {
    return EdgeInsets.symmetric(horizontal: 60, vertical: 30);
  } else if (width > 600) {
    return EdgeInsets.symmetric(horizontal: 40, vertical: 20);
  } else {
    return EdgeInsets.symmetric(horizontal: 20, vertical: 15);
  }
}

@override
Widget build(BuildContext context) {
  final screenWidth = MediaQuery.of(context).size.width;
  final isMobile = screenWidth < 600;
  final isTablet = screenWidth >= 600 && screenWidth < 1200;

  return GestureDetector(
    onTap: () => FocusManager.instance.primaryFocus?.unfocus(),
    child: Scaffold(
      key: scaffoldKey,
      backgroundColor: FlutterFlowTheme.of(context).primaryBackground,
      body: SafeArea(
        child: Column(
          children: [

            Material(
              color: Colors.transparent,
              elevation: 2.0,
              child: Container(
                width: double.infinity,
                decoration: BoxDecoration(
                  color: FlutterFlowTheme.of(context).secondaryBackground,
                ),
                child: wrapWithModel(
                  model: _model.appBarModel,
                  updateCallback: () => safeSetState(() {}),
                  child: AppBarWidget(
                    drawer: () => scaffoldKey.currentState!.openDrawer(),
                  ),
                ),
              ),
            ),


            Expanded(
              child: Row(
                crossAxisAlignment: CrossAxisAlignment.start,
```

```
                children: [

                  if (!isMobile) wrapWithModel(
                    model: _model.sideNavModel,
                    updateCallback: () => safeSetState(() {}),
                    child: SideNavWidget(),
                  ),


                  Expanded(
                    child: SingleChildScrollView(
                      child: Padding(
                        padding: responsivePadding(context),
                        child: Column(
                          crossAxisAlignment: CrossAxisAlignment.start,
                          children: [

                            Text(
                              'Settings',
                              style: FlutterFlowTheme.of(context)
                                  .bodyMedium
                                  .override(
                                fontSize: responsiveFontSize(context,
                                    desktop: 42, tablet: 36, mobile: 28),
                                fontWeight: FontWeight.w600,
                              ),
                            ),
                            SizedBox(height: isMobile ? 8 : 16),
                            Text(
                              'Manage your settings and preferences',
                              style: FlutterFlowTheme.of(context)
                                  .bodyMedium
                                  .override(
                                fontSize: responsiveFontSize(context,
                                    desktop: 20, tablet: 18, mobile: 16),
                              ),
                            ),
                            SizedBox(height: isMobile ? 16 : 30),


                            _buildSettingsItem(
                              context,
                              icon: Icons.nights_stay,
                              title: 'Dark/Light Mode',
                              description: 'Toggle dark theme Appearance',
                              trailing: Switch.adaptive(
                                value: _model.switchValue1!,
                                onChanged: (newValue) =>
```

```
                                    safeSetState(() => _model.switchValue1 =
newValue),
                                activeColor: Color(0xFF203B7E),
                              ),
                            ),

                            _buildSettingsItem(
                              context,
                              icon: Icons.payment,
                              title: 'Payment Method',
                              description: 'Add your account details',
                              trailing:
Icon(Icons.arrow_forward_ios_rounded),
                            ),

                            _buildSettingsItem(
                              context,
                              icon: Icons.notifications_outlined,
                              title: 'Notifications',
                              description: 'Toggle to turn off the
notifications',
                              trailing: Switch.adaptive(
                                value: _model.switchValue2!,
                                onChanged: (newValue) =>
                                    safeSetState(() => _model.switchValue2 =
newValue),
                                activeColor: Color(0xFF203B7E),
                              ),
                            ),

                            _buildSettingsItem(
                              context,
                              icon: FontAwesomeIcons.fileAlt,
                              title: 'Terms and Privacy Policies',
                              description: 'Read terms and privacy policies',
                              onTap: () => context.pushNamed(
                                  TermsAndConditionWidget.routeName),
                              trailing:
Icon(Icons.arrow_forward_ios_rounded),
                            ),


                            _buildSettingsItem(
                              context,
                              icon: Icons.language_sharp,
                              title: 'Language',
                              description: 'Choose your prefer language',
                              trailing: Row(
                                children: [
```

```
                                    _buildLanguageOption(context, 'French',
isSelected: false),
                                    SizedBox(width: 8),
                                    _buildLanguageOption(context, 'English',
isSelected: true),
                                  ],
                                ),
                              ),

                              _buildSettingsItem(
                                context,
                                icon: Icons.person_off,
                                title: 'Account Deactivation',
                                description: 'Deactivate your account',
                                trailing:
Icon(Icons.arrow_forward_ios_rounded),
                              ),


                              _buildSettingsItem(
                                context,
                                icon: Icons.logout,
                                title: 'Log Out',
                                description: 'Sign out of your account',
                                onTap: () async {
                                  GoRouter.of(context).prepareAuthEvent();
                                  await authManager.signOut();
                                  GoRouter.of(context).clearRedirectLocation();
                                  context.pushNamedAuth(
                                      LandingPageWidget.routeName,
                                      context.mounted);
                                },
                              ),
                            ],
                          ),
                        ),
                      ),
                    ),
                  ),
                ],
              ),
            ),
          ),
        ),
      ),
    );
  }
```

```
Widget _buildSettingsItem(
    BuildContext context, {
      required IconData icon,
      required String title,
      required String description,
      Widget? trailing,
      VoidCallback? onTap,
    }) {
  final isMobile = MediaQuery.of(context).size.width < 600;

  return Padding(
    padding: EdgeInsets.only(bottom: isMobile ? 12 : 17),
    child: Material(
      color: Colors.transparent,
      borderRadius: BorderRadius.circular(18),
      child: InkWell(
        borderRadius: BorderRadius.circular(18),
        onTap: onTap,
        child: Container(
          padding: EdgeInsets.all(isMobile ? 12 : 16),
          decoration: BoxDecoration(
            color: FlutterFlowTheme.of(context).primaryBackground,
            borderRadius: BorderRadius.circular(18),
          ),
          child: Row(
            children: [
              Icon(
                icon,
                color: FlutterFlowTheme.of(context).primaryText,
                size: isMobile ? 40 : 50,
              ),
              SizedBox(width: isMobile ? 8 : 12),
              Expanded(
                child: Column(
                  crossAxisAlignment: CrossAxisAlignment.start,
                  children: [
                    Text(
                      title,
                      style: FlutterFlowTheme.of(context)
                          .bodyMedium
                          .override(
                        fontSize: responsiveFontSize(context,
                            desktop: 22, tablet: 20, mobile: 18),
                        fontWeight: FontWeight.w600,
                      ),
                    ),
                    SizedBox(height: 4),
                    Text(
                      description,
```

```
                      style: FlutterFlowTheme.of(context)
                          .bodyMedium
                          .override(
                        fontSize: responsiveFontSize(context,
                            desktop: 16, tablet: 14, mobile: 12),
                      ),
                    ),
                  ],
                ),
              ),
              if (trailing != null) trailing,
            ],
          ),
        ),
      ),
    ),
  );
 }


 Widget _buildLanguageOption(BuildContext context, String text, {bool
isSelected = false}) {
  return Container(
    width: 80,
    height: 40,
    decoration: BoxDecoration(
      color: isSelected
          ? Color(0xFFD9D9D9)
          : FlutterFlowTheme.of(context).secondaryBackground,
      borderRadius: BorderRadius.circular(8),
    ),
    child: Center(
      child: Text(
        text,
        style: FlutterFlowTheme.of(context)
            .bodyMedium
            .override(
          fontSize: 16,
          fontWeight: FontWeight.bold,
        ),
      ),
    ),
  );
 }
}import '/auth/firebase_auth/auth_util.dart';
import '/componenet/app_bar/app_bar_widget.dart';
import '/componenet/side_nav/side_nav_widget.dart';
import '/flutter_flow/flutter_flow_theme.dart';
import '/flutter_flow/flutter_flow_util.dart';
```

```dart
import '/index.dart';
import 'package:flutter/material.dart';
import 'package:font_awesome_flutter/font_awesome_flutter.dart';
import 'settings_model.dart';
export 'settings_model.dart';

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

 @override
 State<SettingsWidget> createState() => _SettingsWidgetState();
}

class _SettingsWidgetState extends State<SettingsWidget> {
 late SettingsModel _model;
 final scaffoldKey = GlobalKey<ScaffoldState>();

 @override
 void initState() {
   super.initState();
   _model = createModel(context, () => SettingsModel());
   _model.switchValue1 = true;
   _model.switchValue2 = true;
   WidgetsBinding.instance.addPostFrameCallback((_) => safeSetState(() {}));
 }

 @override
 void dispose() {
   _model.dispose();
   super.dispose();
 }


 double responsiveFontSize(BuildContext context,
     {double desktop = 22, double tablet = 20, double mobile = 18}) {
   final width = MediaQuery.of(context).size.width;
   if (width > 1200) return desktop;
   if (width > 600) return tablet;
   return mobile;
 }


 EdgeInsets responsivePadding(BuildContext context) {
   final width = MediaQuery.of(context).size.width;
   if (width > 1200) {
     return EdgeInsets.symmetric(horizontal: 60, vertical: 30);
   } else if (width > 600) {
     return EdgeInsets.symmetric(horizontal: 40, vertical: 20);
   } else {
```

```
    return EdgeInsets.symmetric(horizontal: 20, vertical: 15);
  }
}

@override
Widget build(BuildContext context) {
  final screenWidth = MediaQuery.of(context).size.width;
  final isMobile = screenWidth < 600;
  final isTablet = screenWidth >= 600 && screenWidth < 1200;

  return GestureDetector(
    onTap: () => FocusManager.instance.primaryFocus?.unfocus(),
    child: Scaffold(
      key: scaffoldKey,
      backgroundColor: FlutterFlowTheme.of(context).primaryBackground,
      body: SafeArea(
        child: Column(
          children: [

            Material(
              color: Colors.transparent,
              elevation: 2.0,
              child: Container(
                width: double.infinity,
                decoration: BoxDecoration(
                  color: FlutterFlowTheme.of(context).secondaryBackground,
                ),
                child: wrapWithModel(
                  model: _model.appBarModel,
                  updateCallback: () => safeSetState(() {}),
                  child: AppBarWidget(
                    drawer: () => scaffoldKey.currentState!.openDrawer(),
                  ),
                ),
              ),
            ),


            Expanded(
              child: Row(
                crossAxisAlignment: CrossAxisAlignment.start,
                children: [

                  if (!isMobile) wrapWithModel(
                    model: _model.sideNavModel,
                    updateCallback: () => safeSetState(() {}),
                    child: SideNavWidget(),
                  ),
```

```
                  Expanded(
                    child: SingleChildScrollView(
                      child: Padding(
                        padding: responsivePadding(context),
                        child: Column(
                          crossAxisAlignment: CrossAxisAlignment.start,
                          children: [

                            Text(
                              'Settings',
                              style: FlutterFlowTheme.of(context)
                                  .bodyMedium
                                  .override(
                                fontSize: responsiveFontSize(context,
                                    desktop: 42, tablet: 36, mobile: 28),
                                fontWeight: FontWeight.w600,
                              ),
                            ),
                            SizedBox(height: isMobile ? 8 : 16),
                            Text(
                              'Manage your settings and preferences',
                              style: FlutterFlowTheme.of(context)
                                  .bodyMedium
                                  .override(
                                fontSize: responsiveFontSize(context,
                                    desktop: 20, tablet: 18, mobile: 16),
                              ),
                            ),
                            SizedBox(height: isMobile ? 16 : 30),


                            _buildSettingsItem(
                              context,
                              icon: Icons.nights_stay,
                              title: 'Dark/Light Mode',
                              description: 'Toggle dark theme Appearance',
                              trailing: Switch.adaptive(
                                value: _model.switchValue1!,
                                onChanged: (newValue) =>
                                    safeSetState(() => _model.switchValue1 =
newValue),

                                activeColor: Color(0xFF203B7E),
                              ),
                            ),

                            _buildSettingsItem(
                              context,
                              icon: Icons.payment,
```

```
                              title: 'Payment Method',
                              description: 'Add your account details',
                              trailing:
Icon(Icons.arrow_forward_ios_rounded),
                            ),

                            _buildSettingsItem(
                              context,
                              icon: Icons.notifications_outlined,
                              title: 'Notifications',
                              description: 'Toggle to turn off the
notifications',
                              trailing: Switch.adaptive(
                                value: _model.switchValue2!,
                                onChanged: (newValue) =>
                                    safeSetState(() => _model.switchValue2 =
newValue),
                                activeColor: Color(0xFF203B7E),
                              ),
                            ),

                            _buildSettingsItem(
                              context,
                              icon: FontAwesomeIcons.fileAlt,
                              title: 'Terms and Privacy Policies',
                              description: 'Read terms and privacy policies',
                              onTap: () => context.pushNamed(
                                  TermsAndConditionWidget.routeName),
                              trailing:
Icon(Icons.arrow_forward_ios_rounded),
                            ),


                            _buildSettingsItem(
                              context,
                              icon: Icons.language_sharp,
                              title: 'Language',
                              description: 'Choose your prefer language',
                              trailing: Row(
                                children: [
                                  _buildLanguageOption(context, 'French',
isSelected: false),
                                  SizedBox(width: 8),
                                  _buildLanguageOption(context, 'English',
isSelected: true),
                                ],
                              ),
                            ),
```

```
                            _buildSettingsItem(
                              context,
                              icon: Icons.person_off,
                              title: 'Account Deactivation',
                              description: 'Deactivate your account',
                              trailing:
Icon(Icons.arrow_forward_ios_rounded),
                            ),


                            _buildSettingsItem(
                              context,
                              icon: Icons.logout,
                              title: 'Log Out',
                              description: 'Sign out of your account',
                              onTap: () async {
                                GoRouter.of(context).prepareAuthEvent();
                                await authManager.signOut();
                                GoRouter.of(context).clearRedirectLocation();
                                context.pushNamedAuth(
                                    LandingPageWidget.routeName,
                                    context.mounted);
                              },
                            ),
                          ],
                        ),
                      ),
                    ),
                  ),
                ],
              ),
            ),
          ],
        ),
      ),
    ),
  );
}


Widget _buildSettingsItem(
    BuildContext context, {
      required IconData icon,
      required String title,
      required String description,
      Widget? trailing,
      VoidCallback? onTap,
    }) {
  final isMobile = MediaQuery.of(context).size.width < 600;
```

```
return Padding(
  padding: EdgeInsets.only(bottom: isMobile ? 12 : 17),
  child: Material(
    color: Colors.transparent,
    borderRadius: BorderRadius.circular(18),
    child: InkWell(
      borderRadius: BorderRadius.circular(18),
      onTap: onTap,
      child: Container(
        padding: EdgeInsets.all(isMobile ? 12 : 16),
        decoration: BoxDecoration(
          color: FlutterFlowTheme.of(context).primaryBackground,
          borderRadius: BorderRadius.circular(18),
        ),
        child: Row(
          children: [
            Icon(
              icon,
              color: FlutterFlowTheme.of(context).primaryText,
              size: isMobile ? 40 : 50,
            ),
            SizedBox(width: isMobile ? 8 : 12),
            Expanded(
              child: Column(
                crossAxisAlignment: CrossAxisAlignment.start,
                children: [
                  Text(
                    title,
                    style: FlutterFlowTheme.of(context)
                        .bodyMedium
                        .override(
                      fontSize: responsiveFontSize(context,
                          desktop: 22, tablet: 20, mobile: 18),
                      fontWeight: FontWeight.w600,
                    ),
                  ),
                  SizedBox(height: 4),
                  Text(
                    description,
                    style: FlutterFlowTheme.of(context)
                        .bodyMedium
                        .override(
                      fontSize: responsiveFontSize(context,
                          desktop: 16, tablet: 14, mobile: 12),
                    ),
                  ),
                ],
              ),
```

```
              ),
              if (trailing != null) trailing,
            ],
          ),
        ),
      ),
    ),
  );
 }


 Widget _buildLanguageOption(BuildContext context, String text, {bool
isSelected = false}) {
   return Container(
     width: 80,
     height: 40,
     decoration: BoxDecoration(
       color: isSelected
           ? Color(0xFFD9D9D9)
           : FlutterFlowTheme.of(context).secondaryBackground,
       borderRadius: BorderRadius.circular(8),
     ),
     child: Center(
       child: Text(
         text,
         style: FlutterFlowTheme.of(context)
             .bodyMedium
             .override(
           fontSize: 16,
           fontWeight: FontWeight.bold,
         ),
       ),
     ),
   );
 }
}
```

Profile Responsive code

```
import '/auth/firebase_auth/auth_util.dart';
import '/componenet/app_bar/app_bar_widget.dart';
import '/componenet/data_table/data_table_widget.dart';
import '/componenet/reviews/reviews_widget.dart';
import '/componenet/side_nav/side_nav_widget.dart';
import '/flutter_flow/flutter_flow_theme.dart';
import '/flutter_flow/flutter_flow_util.dart';
```

```dart
import '/flutter_flow/flutter_flow_widgets.dart';
import 'package:auto_size_text/auto_size_text.dart';
import 'package:flutter/material.dart';
import 'profile_model.dart';
export 'profile_model.dart';

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

 @override
 State<ProfileWidget> createState() => _ProfileWidgetState();
}

class _ProfileWidgetState extends State<ProfileWidget>
   with TickerProviderStateMixin {
 late ProfileModel _model;
 final scaffoldKey = GlobalKey<ScaffoldState>();

 @override
 void initState() {
   super.initState();
   _model = createModel(context, () => ProfileModel());
   _model.tabBarController = TabController(
     vsync: this,
     length: 5,
     initialIndex: 0,
   )..addListener(() => safeSetState(() {}));
 }

 @override
 void dispose() {
   _model.dispose();
   super.dispose();
 }

 double _responsiveFontSize(BuildContext context) {
   final width = MediaQuery.of(context).size.width;
   if (width > 1200) return 32;
   if (width > 600) return 28;
   return 24;
 }

 EdgeInsets _responsivePadding(BuildContext context) {
   final width = MediaQuery.of(context).size.width;
   if (width > 1200) return EdgeInsets.symmetric(horizontal: 40, vertical:
20);
   if (width > 600) return EdgeInsets.symmetric(horizontal: 20, vertical: 15);
   return EdgeInsets.symmetric(horizontal: 16, vertical: 10);
 }
```

```dart
@override
Widget build(BuildContext context) {
  final screenWidth = MediaQuery.of(context).size.width;
  final isMobile = screenWidth < 600;
  final isTablet = screenWidth >= 600 && screenWidth < 1200;

  return GestureDetector(
    onTap: () => FocusManager.instance.primaryFocus?.unfocus(),
    child: Scaffold(
      key: scaffoldKey,
      backgroundColor: FlutterFlowTheme.of(context).secondaryBackground,
      body: SafeArea(
        child: Column(
          children: [
            Material(
              color: Colors.transparent,
              elevation: 2.0,
              child: Container(
                width: double.infinity,
                decoration: BoxDecoration(
                  color: FlutterFlowTheme.of(context).secondaryBackground,
                ),
                child: wrapWithModel(
                  model: _model.appBarModel,
                  child: AppBarWidget(
                    drawer: () => scaffoldKey.currentState!.openDrawer(),
                  ),
                ),
              ),
            ),

            Expanded(
              child: Row(
                crossAxisAlignment: CrossAxisAlignment.start,
                children: [
                  if (!isMobile) wrapWithModel(
                    model: _model.sideNavModel,
                    child: SideNavWidget(),
                  ),

                  Expanded(
                    child: SingleChildScrollView(
                      child: Padding(
                        padding: _responsivePadding(context),
                        child: Column(
                          crossAxisAlignment: CrossAxisAlignment.start,
                          children: [
                            _buildProfileHeader(context, isMobile),
```

```
                          _buildBioSection(context, isMobile),

                          _buildSkillsTags(context, isMobile),

                          _buildTabContent(context, isMobile),
                        ],
                      ),
                    ),
                  ),
                ),
              ],
            ),
          ),
        ),
      ),
    );
  }

  Widget _buildProfileHeader(BuildContext context, bool isMobile) {
    final fontSize = _responsiveFontSize(context);

    return Row(
      mainAxisAlignment: MainAxisAlignment.spaceBetween,
      crossAxisAlignment: CrossAxisAlignment.start,
      children: [
        Flexible(
          child: Row(
            crossAxisAlignment: CrossAxisAlignment.start,
            children: [
              AuthUserStreamWidget(
                builder: (context) => ClipRRect(
                  borderRadius: BorderRadius.circular(
                    valueOrDefault(currentUserDocument?.currentRole, '') ==
'Client' ? 0.0 : 250.0,
                  ),
                  child: Image.network(
                    'https://picsum.photos/seed/658/600',
                    width: isMobile ? 120 : 192,
                    height: isMobile ? 120 : 192,
                    fit: BoxFit.cover,
                  ),
                ),
              ),

              SizedBox(width: isMobile ? 12 : 16),
```

```
            Flexible(
              child: Column(
                crossAxisAlignment: CrossAxisAlignment.start,
                children: [
                  Text(
                    'Jeniffer Wington',
                    style: FlutterFlowTheme.of(context).bodyMedium.override(
                      fontSize: fontSize,
                      fontWeight: FontWeight.bold,
                    ),
                  ),

                  _buildInfoRow(
                    context,
                    icon: Icons.location_on_outlined,
                    text: 'Calgary, AB, Canada',
                    fontSize: fontSize - 4,
                  ),

                  if (!isMobile) _buildUserMetadata(context, fontSize),
                ],
              ),
            ),
          ],
        ),
      ),

      if (!isMobile) FFButtonWidget(
        onPressed: () => print('Edit Profile'),
        text: 'Edit Profile',
        options: FFButtonOptions(
          width: 164,
          height: 43,
          padding: EdgeInsets.symmetric(horizontal: 16),
          color: Color(0xFFDDDEF4),
          textStyle: FlutterFlowTheme.of(context).titleSmall.override(
            fontSize: 20,
            color: Color(0xFF343B8C),
          ),
          borderRadius: BorderRadius.circular(18),
        ),
      ),
    ],
  );
}

Widget _buildInfoRow(BuildContext context, {required IconData icon, required
String text, required double fontSize}) {
  return Padding(
```

```
    padding: EdgeInsets.only(top: 4),
    child: Row(
      children: [
        Icon(icon, color: FlutterFlowTheme.of(context).secondaryText, size:
fontSize),
        SizedBox(width: 4),
        Text(
          text,
          style: FlutterFlowTheme.of(context).bodyMedium.override(
            fontSize: fontSize,
            color: FlutterFlowTheme.of(context).secondaryText,
          ),
        ),
      ],
    ),
  );
}

Widget _buildUserMetadata(BuildContext context, double fontSize) {
  return Row(
    children: [
      _buildInfoRow(
        context,
        icon: Icons.star_border,
        text: '4.5 (27 Reviews)',
        fontSize: fontSize - 4,
      ),

      SizedBox(width: 25),

      _buildInfoRow(
        context,
        icon: Icons.verified_sharp,
        text: 'Verified',
        fontSize: fontSize - 4,
      ),

      SizedBox(width: 25),

      _buildInfoRow(
        context,
        icon: Icons.calendar_today_outlined,
        text: 'Member since Jan 2024',
        fontSize: fontSize - 4,
      ),
    ],
  );
}
```

```
Widget _buildBioSection(BuildContext context, bool isMobile) {
  return Padding(
    padding: EdgeInsets.only(top: isMobile ? 12 : 20),
    child: AutoSizeText(
      'I regularly hire skilled professionals for various tasks...',
      minFontSize: 16,
      style: FlutterFlowTheme.of(context).bodyMedium.override(
        fontSize: isMobile ? 18 : 22,
        color: FlutterFlowTheme.of(context).secondaryText,
        fontWeight: FontWeight.w500,
      ),
    ),
  );
}

Widget _buildSkillsTags(BuildContext context, bool isMobile) {
  final skills = ['Interior Design', 'Home Decor', 'Furniture Layout', 'Wall
Art'];

  return Padding(
    padding: EdgeInsets.only(top: isMobile ? 12 : 20),
    child: Wrap(
      spacing: isMobile ? 8 : 21,
      runSpacing: isMobile ? 8 : 18,
      children: [
        ...skills.map((skill) => FFButtonWidget(
          onPressed: () => print('$skill pressed'),
          text: skill,
          options: FFButtonOptions(
            height: 43,
            padding: EdgeInsets.symmetric(horizontal: 16),
            color: Color(0xFFDDDEF4),
            textStyle: FlutterFlowTheme.of(context).titleSmall.override(
              fontSize: isMobile ? 16 : 20,
              color: Color(0xFF343B8C),
            ),
            borderRadius: BorderRadius.circular(18),
          ),
        )),

        if (!isMobile) Text(
          'View all',
          style: FlutterFlowTheme.of(context).bodyMedium.override(
            fontSize: 24,
            color: FlutterFlowTheme.of(context).secondaryText,
            decoration: TextDecoration.underline,
          ),
        ),
      ],
```

```
    ),
  );
}

Widget _buildTabContent(BuildContext context, bool isMobile) {
  return Expanded(
    child: Column(
      children: [
        Align(
          alignment: Alignment.center,
          child: TabBar(
            isScrollable: isMobile, // Allow scrolling on mobile
            labelColor: Color(0xFFF21A1A),
            unselectedLabelColor: FlutterFlowTheme.of(context).primaryText,
            labelStyle: FlutterFlowTheme.of(context).titleMedium.override(
              fontSize: isMobile ? 24 : 32,
              fontWeight: FontWeight.w500,
            ),
            indicatorColor: Color(0xFF343B8C),
            tabs: [
              Tab(text: 'All Jobs'),
              Tab(text: 'Active Jobs'),
              Tab(text: 'Complete Jobs'),
              Tab(text: 'Drafts'),
              Tab(text: 'Reviews'),
            ],
            controller: _model.tabBarController,
          ),
        ),

        Expanded(
          child: TabBarView(
            controller: _model.tabBarController,
            children: [
              DataTableWidget(),

              DataTableWidget(),

              DataTableWidget(),

              DataTableWidget(),

              _buildReviewsSection(isMobile),
            ],
          ),
        ),
      ],
    ),
  );
```

```
 }

 Widget _buildReviewsSection(bool isMobile) {
   return Padding(
     padding: EdgeInsets.symmetric(horizontal: isMobile ? 8 : 16),
     child: LayoutBuilder(
       builder: (context, constraints) {
         final crossAxisCount = constraints.maxWidth > 800 ? 3
             : constraints.maxWidth > 500 ? 2
             : 1;

         return GridView.count(
           crossAxisCount: crossAxisCount,
           childAspectRatio: 3/2,
           crossAxisSpacing: isMobile ? 8 : 25,
           mainAxisSpacing: isMobile ? 8 : 16,
           children: List.generate(3, (index) =>
               wrapWithModel(
                 model: _model.reviewsModels[index],
                 child: ReviewsWidget(),
               ),
           ),
         );
       },
     ),
   );
 }
}
```

Chats Responsive code

```
import '/componenet/app_bar/app_bar_widget.dart';
import '/componenet/chat_bubbles/chat_bubbles_widget.dart';
import '/componenet/side_nav/side_nav_widget.dart';
import '/flutter_flow/flutter_flow_theme.dart';
import '/flutter_flow/flutter_flow_util.dart';
import '/flutter_flow/flutter_flow_widgets.dart';
import 'package:flutter/material.dart';
import 'chats_model.dart';
export 'chats_model.dart';

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

 @override
 State<ChatsWidget> createState() => _ChatsWidgetState();
}

class _ChatsWidgetState extends State<ChatsWidget> {
```

```
late ChatsModel _model;
final scaffoldKey = GlobalKey<ScaffoldState>();

@override
void initState() {
  super.initState();
  _model = createModel(context, () => ChatsModel());
  _model.textController ??= TextEditingController();
  _model.textFieldFocusNode ??= FocusNode();
}

@override
void dispose() {
  _model.dispose();
  super.dispose();
}

double _responsiveFontSize(BuildContext context) {
  final width = MediaQuery.of(context).size.width;
  if (width > 1200) return 22;
  if (width > 600) return 20;
  return 18;
}

EdgeInsets _responsivePadding(BuildContext context) {
  final width = MediaQuery.of(context).size.width;
  if (width > 1200) return EdgeInsets.symmetric(horizontal: 20, vertical:
10);
  if (width > 600) return EdgeInsets.symmetric(horizontal: 15, vertical: 8);
  return EdgeInsets.symmetric(horizontal: 10, vertical: 5);
}

double _responsiveImageSize(BuildContext context) {
  final width = MediaQuery.of(context).size.width;
  if (width > 1200) return 88;
  if (width > 600) return 72;
  return 60;
}

@override
Widget build(BuildContext context) {
  final screenWidth = MediaQuery.of(context).size.width;
  final isMobile = screenWidth < 600;
  final isTablet = screenWidth >= 600 && screenWidth < 1200;

  return GestureDetector(
    onTap: () => FocusManager.instance.primaryFocus?.unfocus(),
    child: Scaffold(
      key: scaffoldKey,
```

```
      backgroundColor: FlutterFlowTheme.of(context).secondaryBackground,
      body: SafeArea(
        child: Column(
          children: [
            Material(
              color: Colors.transparent,
              elevation: 2.0,
              child: Container(
                width: double.infinity,
                decoration: BoxDecoration(
                  color: FlutterFlowTheme.of(context).secondaryBackground,
                ),
                child: wrapWithModel(
                  model: _model.appBarModel,
                  child: AppBarWidget(
                    drawer: () => scaffoldKey.currentState!.openDrawer(),
                  ),
                ),
              ),
            ),

            Expanded(
              child: Row(
                crossAxisAlignment: CrossAxisAlignment.start,
                children: [
                  if (!isMobile) wrapWithModel(
                    model: _model.sideNavModel,
                    child: SideNavWidget(),
                  ),

                  if (!isMobile) Container(
                    width: isTablet ? screenWidth * 0.35 : screenWidth *
0.25,
                    child: _buildChatList(context, isMobile),
                  ),

                  // Vertical Divider - Only show on larger screens
                  if (!isMobile) SizedBox(
                    height: double.infinity,
                    child: VerticalDivider(
                      thickness: 2.0,
                      color: FlutterFlowTheme.of(context).primaryText,
                    ),
                  ),

                  // Main Chat Area
                  Expanded(
                      child: _buildMainChatArea(context, isMobile),
```

```
                    // Order Info Panel - Hidden on mobile and tablet
                    if (!isMobile && !isTablet) SizedBox(
                  height: double.infinity,
                  child: VerticalDivider(
                    thickness: 2.0,
                    color: FlutterFlowTheme.of(context).primaryText,
                  ),
                ),

                if (!isMobile && !isTablet) Container(
                  width: 220,
                  child: _buildOrderInfoPanel(context),
                ),
              ],
            ),
          ),
        ],
      ),
    ),
  );
}

Widget _buildChatList(BuildContext context, bool isMobile) {
  return Column(
    children: [
      Padding(
        padding: _responsivePadding(context),
        child: Align(
          alignment: Alignment.centerLeft,
          child: Text(
            'Messages',
            style: FlutterFlowTheme.of(context).bodyMedium.override(
              fontSize: _responsiveFontSize(context) + 10,
              fontWeight: FontWeight.w600,
            ),
          ),
        ),
      ),
      Expanded(
        child: ListView.builder(
          padding: EdgeInsets.only(bottom: 80),
          itemCount: 5, // Replace with actual chat count
          itemBuilder: (context, index) => _buildChatListItem(context,
index),
        ),
      ),
    ],
  );
```

```
}

Widget _buildChatListItem(BuildContext context, int index) {
  final imageSize = _responsiveImageSize(context);

  return Container(
    decoration: BoxDecoration(
      color: FlutterFlowTheme.of(context).primaryBackground,
    ),
    child: Padding(
      padding: _responsivePadding(context),
      child: Row(
        mainAxisAlignment: MainAxisAlignment.spaceBetween,
        children: [
          Row(
            children: [
              Container(
                width: imageSize,
                height: imageSize,
                clipBehavior: Clip.antiAlias,
                decoration: BoxDecoration(
                  shape: BoxShape.circle,
                ),
                child: Image.network(
                  'https://picsum.photos/seed/361/600',
                  fit: BoxFit.cover,
                ),
              ),
              SizedBox(width: 12),
              Column(
                crossAxisAlignment: CrossAxisAlignment.start,
                children: [
                  Text(
                    'Sara Kim',
                    style: FlutterFlowTheme.of(context).bodyMedium.override(
                      fontSize: _responsiveFontSize(context),
                      fontWeight: FontWeight.w600,
                    ),
                  ),
                  Text(
                    'Its pleasure doing wo',
                    maxLines: 1,
                    style: FlutterFlowTheme.of(context).bodyMedium.override(
                      fontSize: _responsiveFontSize(context) - 4,
                      fontWeight: FontWeight.w500,
                    ),
                  ),
                ],
              ),
```

```
            ],
          ),
          Text(
            '11:45 am',
            style: FlutterFlowTheme.of(context).bodyMedium.override(
              fontSize: _responsiveFontSize(context) - 4,
              fontWeight: FontWeight.w500,
            ),
          ),
        ],
      ),
    ),
  );
}

Widget _buildMainChatArea(BuildContext context, bool isMobile) {
  final imageSize = _responsiveImageSize(context);

  return Column(
      children: [
      // Chat Header
      Padding(
      padding: _responsivePadding(context),
  child: Row(
  mainAxisAlignment: MainAxisAlignment.spaceBetween,
  children: [
  Row(
  children: [
  Container(
  width: imageSize,
  height: imageSize,
  clipBehavior: Clip.antiAlias,
  decoration: BoxDecoration(
  shape: BoxShape.circle,
  ),
  child: Image.network(
  'https://picsum.photos/seed/295/600',
  fit: BoxFit.cover,
  ),
  ),
  SizedBox(width: 12),
  Column(
  crossAxisAlignment: CrossAxisAlignment.start,
  children: [
  Text(
  'Sara Kim',
  style: FlutterFlowTheme.of(context).bodyMedium.override(
  fontSize: _responsiveFontSize(context),
  fontWeight: FontWeight.w600,
```

```
),
),
Text(
'Its pleasure doing wo',
maxLines: 1,
style: FlutterFlowTheme.of(context).bodyMedium.override(
fontSize: _responsiveFontSize(context) - 4,
fontWeight: FontWeight.w500,
),
),
],
),
],
),
if (!isMobile) FFButtonWidget(
onPressed: () => print('View Job'),
text: 'View Job',
options: FFButtonOptions(
height: 40,
padding: EdgeInsets.symmetric(horizontal: 16),
color: Color(0xFFE7E9EE),
textStyle: FlutterFlowTheme.of(context).titleSmall.override(
fontSize: _responsiveFontSize(context),
color: Color(0xFF203B7E),
borderRadius: BorderRadius.circular(20),
),
),
],
),
),

Divider(
thickness: 2.0,
color: FlutterFlowTheme.of(context).primaryText,
),

Expanded(
child: Padding(
padding: EdgeInsets.only(bottom: 10),
child: ListView(
reverse: true,
children: [
wrapWithModel(
model: _model.chatBubblesModel,
child: ChatBubblesWidget(
parameter1: _model.show,
),
),
],
```

```
),
),
),

Padding(
padding: _responsivePadding(context),
child: Row(
children: [
Expanded(
child: Container(
height: 65,
decoration: BoxDecoration(
color: Color(0xFFF0EDED),
borderRadius: BorderRadius.circular(30),
child: TextFormField(
controller: _model.textController,
focusNode: _model.textFieldFocusNode,
decoration: InputDecoration(
hintText: 'Type your message here',
hintStyle: FlutterFlowTheme.of(context).labelMedium.override(
fontSize: _responsiveFontSize(context),
),
border: InputBorder.none,
contentPadding: EdgeInsets.symmetric(horizontal: 20),
),
style: FlutterFlowTheme.of(context).bodyMedium.override(
fontSize: _responsiveFontSize(context),
),
),
),
),
),
SizedBox(width: 10),
FFButtonWidget(
onPressed: () => print('Send'),
text: 'Send',
options: FFButtonOptions(
width: isMobile ? 100 : 131,
height: 50,
color: Color(0xFFF21A1A),
textStyle: FlutterFlowTheme.of(context).titleSmall.override(
fontSize: _responsiveFontSize(context),
color: Colors.white),
borderRadius: BorderRadius.circular(10),
),
),
],
),
),
],
```

```
  );
}

Widget _buildOrderInfoPanel(BuildContext context) {
  return Padding(
    padding: EdgeInsets.only(top: 45),
    child: SingleChildScrollView(
      child: Column(
        children: [
          Text(
            'Orders with You',
            style: FlutterFlowTheme.of(context).bodyMedium.override(
              fontSize: _responsiveFontSize(context),
              fontWeight: FontWeight.w600,
            ),
          ),
          Padding(
            padding: EdgeInsets.only(top: 5, bottom: 15),
            child: Text(
              'Total 05',
              style: FlutterFlowTheme.of(context).bodyMedium.override(
                color: FlutterFlowTheme.of(context).secondaryText,
                fontSize: _responsiveFontSize(context),
              ),
            ),
          ),
          Text(
            'Current Order',
            style: FlutterFlowTheme.of(context).bodyMedium.override(
              fontSize: _responsiveFontSize(context),
              fontWeight: FontWeight.w600,
            ),
          ),
          SizedBox(height: 10),
          ClipRRect(
            borderRadius: BorderRadius.circular(8),
            child: Image.network(
              'https://picsum.photos/seed/773/600',
              width: 200,
              height: 150,
              fit: BoxFit.cover,
            ),
          ),
          Padding(
            padding: EdgeInsets.only(top: 20),
            child: Container(
              width: 200,
              decoration: BoxDecoration(
                color: FlutterFlowTheme.of(context).primaryBackground,
```

```
              ),
              child: Padding(
                padding: EdgeInsets.all(10),
                child: Column(
                  crossAxisAlignment: CrossAxisAlignment.start,
                  children: [
                    _buildOrderDetailRow('Budget', '\$40'),
                    SizedBox(height: 12),
                    _buildOrderDetailRow('Duration', '5 hours'),
                    SizedBox(height: 12),
                    Text(
                      'Job Status :',
                      style:
FlutterFlowTheme.of(context).bodyMedium.override(
                        fontSize: _responsiveFontSize(context),
                        fontWeight: FontWeight.w500,
                      ),
                    ),
                    Align(
                      alignment: Alignment.centerRight,
                      child: Text(
                        'In Progress',
                        style:
FlutterFlowTheme.of(context).bodyMedium.override(
                          color: Colors.green,
                          fontSize: _responsiveFontSize(context),
                          fontWeight: FontWeight.w500,
                        ),
                      ),
                    ),
                  ],
                ),
              ),
            ),
          ),
          Padding(
            padding: EdgeInsets.only(top: 30),
            child: FFButtonWidget(
              onPressed: () => print('Mark as Complete'),
              text: 'Mark as Complete',
              options: FFButtonOptions(
                width: 200,
                height: 54,
                color: Color(0xFFE7E9EE),
                textStyle: FlutterFlowTheme.of(context).titleSmall.override(
                    color: Color(0xFF203B7E),
                    fontSize: _responsiveFontSize(context),
                    fontWeight: FontWeight.w600),
                borderRadius: BorderRadius.circular(16),
```

```
              ),
            ),
          ),
        ],
      ),
    ),
  );
 }

 Widget _buildOrderDetailRow(String label, String value) {
   return Row(
     mainAxisAlignment: MainAxisAlignment.spaceBetween,
     children: [
       Text(
         label,
         style: FlutterFlowTheme.of(context).bodyMedium.override(
           fontSize: _responsiveFontSize(context),
           fontWeight: FontWeight.w500,
         ),
       ),
       Text(
         value,
         style: FlutterFlowTheme.of(context).bodyMedium.override(
           fontSize: _responsiveFontSize(context),
           fontWeight: FontWeight.w500,
         ),
       ),
     ],
   );
 }
}
```

Backend Code

```
import 'package:cloud_firestore/cloud_firestore.dart';
import 'package:firebase_auth/firebase_auth.dart';
import '../auth/firebase_auth/auth_util.dart';
import '../flutter_flow/flutter_flow_util.dart';
import 'schema/util/firestore_util.dart';
import 'schema/users_record.dart';
import 'schema/categories_record.dart';
import 'schema/providers_record.dart';
import 'schema/skills_record.dart';
import 'schema/jobs_record.dart';

export 'dart:async' show StreamSubscription;
export 'package:cloud_firestore/cloud_firestore.dart' hide Order;
```

```dart
export 'package:firebase_core/firebase_core.dart';
export 'schema/index.dart';
export 'schema/util/firestore_util.dart';
export 'schema/util/schema_util.dart';

export 'schema/users_record.dart';
export 'schema/categories_record.dart';
export 'schema/providers_record.dart';
export 'schema/skills_record.dart';
export 'schema/jobs_record.dart';


Future<int> queryUsersRecordCount({
 Query Function(Query)? queryBuilder,
 int limit = -1,
}) =>
   queryCollectionCount(
     UsersRecord.collection,
     queryBuilder: queryBuilder,
     limit: limit,
   );

Stream<List<UsersRecord>> queryUsersRecord({
 Query Function(Query)? queryBuilder,
 int limit = -1,
 bool singleRecord = false,
}) =>
   queryCollection(
     UsersRecord.collection,
     UsersRecord.fromSnapshot,
     queryBuilder: queryBuilder,
     limit: limit,
     singleRecord: singleRecord,
   );

Future<List<UsersRecord>> queryUsersRecordOnce({
 Query Function(Query)? queryBuilder,
 int limit = -1,
 bool singleRecord = false,
}) =>
   queryCollectionOnce(
     UsersRecord.collection,
     UsersRecord.fromSnapshot,
     queryBuilder: queryBuilder,
     limit: limit,
     singleRecord: singleRecord,
   );

Future<int> queryCategoriesRecordCount({
```

```
 Query Function(Query)? queryBuilder,
 int limit = -1,
}) =>
   queryCollectionCount(
     CategoriesRecord.collection,
     queryBuilder: queryBuilder,
     limit: limit,
   );

Stream<List<CategoriesRecord>> queryCategoriesRecord({
 Query Function(Query)? queryBuilder,
 int limit = -1,
 bool singleRecord = false,
}) =>
   queryCollection(
     CategoriesRecord.collection,
     CategoriesRecord.fromSnapshot,
     queryBuilder: queryBuilder,
     limit: limit,
     singleRecord: singleRecord,
   );

Future<List<CategoriesRecord>> queryCategoriesRecordOnce({
 Query Function(Query)? queryBuilder,
 int limit = -1,
 bool singleRecord = false,
}) =>
   queryCollectionOnce(
     CategoriesRecord.collection,
     CategoriesRecord.fromSnapshot,
     queryBuilder: queryBuilder,
     limit: limit,
     singleRecord: singleRecord,
   );

Future<int> queryProvidersRecordCount({
 Query Function(Query)? queryBuilder,
 int limit = -1,
}) =>
   queryCollectionCount(
     ProvidersRecord.collection,
     queryBuilder: queryBuilder,
     limit: limit,
   );

Stream<List<ProvidersRecord>> queryProvidersRecord({
 Query Function(Query)? queryBuilder,
 int limit = -1,
 bool singleRecord = false,
```

```
}) =>
   queryCollection(
     ProvidersRecord.collection,
     ProvidersRecord.fromSnapshot,
     queryBuilder: queryBuilder,
     limit: limit,
     singleRecord: singleRecord,
   );

Future<List<ProvidersRecord>> queryProvidersRecordOnce({
 Query Function(Query)? queryBuilder,
 int limit = -1,
 bool singleRecord = false,
}) =>
   queryCollectionOnce(
     ProvidersRecord.collection,
     ProvidersRecord.fromSnapshot,
     queryBuilder: queryBuilder,
     limit: limit,
     singleRecord: singleRecord,
   );

Future<int> querySkillsRecordCount({
 Query Function(Query)? queryBuilder,
 int limit = -1,
}) =>
   queryCollectionCount(
     SkillsRecord.collection,
     queryBuilder: queryBuilder,
     limit: limit,
   );

Stream<List<SkillsRecord>> querySkillsRecord({
 Query Function(Query)? queryBuilder,
 int limit = -1,
 bool singleRecord = false,
}) =>
   queryCollection(
     SkillsRecord.collection,
     SkillsRecord.fromSnapshot,
     queryBuilder: queryBuilder,
     limit: limit,
     singleRecord: singleRecord,
   );

Future<List<SkillsRecord>> querySkillsRecordOnce({
 Query Function(Query)? queryBuilder,
 int limit = -1,
 bool singleRecord = false,
```

```
}) =>
   queryCollectionOnce(
     SkillsRecord.collection,
     SkillsRecord.fromSnapshot,
     queryBuilder: queryBuilder,
     limit: limit,
     singleRecord: singleRecord,
   );

Future<int> queryJobsRecordCount({
 Query Function(Query)? queryBuilder,
 int limit = -1,
}) =>
   queryCollectionCount(
     JobsRecord.collection,
     queryBuilder: queryBuilder,
     limit: limit,
   );

Stream<List<JobsRecord>> queryJobsRecord({
 Query Function(Query)? queryBuilder,
 int limit = -1,
 bool singleRecord = false,
}) =>
   queryCollection(
     JobsRecord.collection,
     JobsRecord.fromSnapshot,
     queryBuilder: queryBuilder,
     limit: limit,
     singleRecord: singleRecord,
   );

Future<List<JobsRecord>> queryJobsRecordOnce({
 Query Function(Query)? queryBuilder,
 int limit = -1,
 bool singleRecord = false,
}) =>
   queryCollectionOnce(
     JobsRecord.collection,
     JobsRecord.fromSnapshot,
     queryBuilder: queryBuilder,
     limit: limit,
     singleRecord: singleRecord,
   );

Future<int> queryCollectionCount(
   Query collection, {
     Query Function(Query)? queryBuilder,
     int limit = -1,
```

```
  }) {
 final builder = queryBuilder ?? (q) => q;
 var query = builder(collection);
 if (limit > 0) {
   query = query.limit(limit);
 }

 return query.count().get().catchError((err) {
   print('Error querying $collection: $err');
 }).then((value) => value.count!);
}

Stream<List<T>> queryCollection<T>(
   Query collection,
   RecordBuilder<T> recordBuilder, {
     Query Function(Query)? queryBuilder,
     int limit = -1,
     bool singleRecord = false,
   }) {
 final builder = queryBuilder ?? (q) => q;
 var query = builder(collection);
 if (limit > 0 || singleRecord) {
   query = query.limit(singleRecord ? 1 : limit);
 }
 return query.snapshots().handleError((err) {
   print('Error querying $collection: $err');
 }).map((s) => s.docs
     .map(
       (d) => safeGet(
         () => recordBuilder(d),
         (e) => print('Error serializing doc ${d.reference.path}:\n$e'),
   ),
 )
     .where((d) => d != null)
     .map((d) => d!)
     .toList());
}

Future<List<T>> queryCollectionOnce<T>(
   Query collection,
   RecordBuilder<T> recordBuilder, {
     Query Function(Query)? queryBuilder,
     int limit = -1,
     bool singleRecord = false,
   }) {
 final builder = queryBuilder ?? (q) => q;
 var query = builder(collection);
 if (limit > 0 || singleRecord) {
   query = query.limit(singleRecord ? 1 : limit);
```

```
 }
 return query.get().then((s) => s.docs
     .map(
       (d) => safeGet(
         () => recordBuilder(d),
         (e) => print('Error serializing doc ${d.reference.path}:\n$e'),
   ),
 )
     .where((d) => d != null)
     .map((d) => d!)
     .toList());
}

Filter filterIn(String field, List? list) => (list?.isEmpty ?? true)
   ? Filter(field, whereIn: null)
   : Filter(field, whereIn: list);

Filter filterArrayContainsAny(String field, List? list) =>
   (list?.isEmpty ?? true)
       ? Filter(field, arrayContainsAny: null)
       : Filter(field, arrayContainsAny: list);

extension QueryExtension on Query {
 Query whereIn(String field, List? list) => (list?.isEmpty ?? true)
     ? where(field, whereIn: null)
     : where(field, whereIn: list);

 Query whereNotIn(String field, List? list) => (list?.isEmpty ?? true)
     ? where(field, whereNotIn: null)
     : where(field, whereNotIn: list);

 Query whereArrayContainsAny(String field, List? list) =>
     (list?.isEmpty ?? true)
         ? where(field, arrayContainsAny: null)
         : where(field, arrayContainsAny: list);
}

class FFFirestorePage<T> {
 final List<T> data;
 final Stream<List<T>>? dataStream;
 final QueryDocumentSnapshot? nextPageMarker;

 FFFirestorePage(this.data, this.dataStream, this.nextPageMarker);
}

Future<FFFirestorePage<T>> queryCollectionPage<T>(
   Query collection,
   RecordBuilder<T> recordBuilder, {
     Query Function(Query)? queryBuilder,
```

```
    DocumentSnapshot? nextPageMarker,
    required int pageSize,
    required bool isStream,
  }) async {
 final builder = queryBuilder ?? (q) => q;
 var query = builder(collection).limit(pageSize);
 if (nextPageMarker != null) {
   query = query.startAfterDocument(nextPageMarker);
 }
 Stream<QuerySnapshot>? docSnapshotStream;
 QuerySnapshot docSnapshot;
 if (isStream) {
   docSnapshotStream = query.snapshots();
   docSnapshot = await docSnapshotStream.first;
 } else {
   docSnapshot = await query.get();
 }
 final getDocs = (QuerySnapshot s) => s.docs
     .map(
       (d) => safeGet(
         () => recordBuilder(d),
         (e) => print('Error serializing doc ${d.reference.path}:\n$e'),
   ),
 )
     .where((d) => d != null)
     .map((d) => d!)
     .toList();
 final data = getDocs(docSnapshot);
 final dataStream = docSnapshotStream?.map(getDocs);
 final nextPageToken = docSnapshot.docs.isEmpty ? null :
docSnapshot.docs.last;
 return FFFirestorePage(data, dataStream, nextPageToken);
}


Future maybeCreateUser(User user) async {
 final userRecord = UsersRecord.collection.doc(user.uid);
 final userExists = await userRecord.get().then((u) => u.exists);
 if (userExists) {
   currentUserDocument = await UsersRecord.getDocumentOnce(userRecord);
   return;
 }

 final userData = createUsersRecordData(
   email: user.email ??
       FirebaseAuth.instance.currentUser?.email ??
       user.providerData.firstOrNull?.email,
   displayName:
   user.displayName ?? FirebaseAuth.instance.currentUser?.displayName,
```

```
    photoUrl: user.photoURL,
    uid: user.uid,
    phoneNumber: user.phoneNumber,
    createdTime: getCurrentTimestamp,
  );

  await userRecord.set(userData);
  currentUserDocument = UsersRecord.getDocumentFromData(userData, userRecord);
}

Future updateUserDocument({String? email}) async {
  await currentUserDocument?.reference
      .update(createUsersRecordData(email: email));
}



class ProfileService {
  static final FirebaseFirestore _firestore = FirebaseFirestore.instance;

  static Future<Map<String, dynamic>> getCompleteProfile(String userId) async {
    try {
      final userDoc = await _firestore.collection('users').doc(userId).get();
      if (!userDoc.exists) return {};

      final skills = await _getUserSkills(userId);
      final reviews = await _getUserReviews(userId);
      final jobs = await _getUserJobs(userId);

      return {
        ...userDoc.data() ?? {},
        'skills': skills,
        'reviews': reviews,
        'jobs': jobs,
      };
    } catch (e) {
      print('Error getting complete profile: $e');
      return {};
    }
  }

  static Future<Map<String, dynamic>> getMobileProfile(String userId) async {
    try {
      final doc = await _firestore.collection('users').doc(userId).get();
      if (!doc.exists) return {};

      return {
        'name': doc.get('displayName'),
        'image': doc.get('photoUrl'),
```

```
      'rating': doc.get('rating') ?? 0,
      'skills': await _getUserSkills(userId, limit: 3),
      'reviewsCount': await _getReviewsCount(userId),
    };
  } catch (e) {
    print('Error getting mobile profile: $e');
    return {};
  }
}

static Future<void> updateProfileFields({
  required String userId,
  required Map<String, dynamic> updates,
  bool isMobile = false,
}) async {
  try {
    final updateData = isMobile
        ? _filterMobileUpdates(updates)
        : updates;

    await _firestore.collection('users').doc(userId).update({
      ...updateData,
      'lastUpdated': FieldValue.serverTimestamp(),
      'lastDevice': isMobile ? 'mobile' : 'desktop',
    });
  } catch (e) {
    print('Error updating profile: $e');
    rethrow;
  }
}

static Future<List<String>> _getUserSkills(String userId, {int? limit}) async
{
  final query =
_firestore.collection('users').doc(userId).collection('skills');
  final snapshot = limit != null ? await query.limit(limit).get() : await
query.get();
  return snapshot.docs.map((doc) => doc.id).toList();
}

static Future<List<Map<String, dynamic>>> _getUserReviews(String userId)
async {
  final snapshot = await _firestore
      .collection('reviews')
      .where('userId', isEqualTo: userId)
      .limit(5)
      .get();
  return snapshot.docs.map((doc) => doc.data()).toList();
}
```

```
 static Future<List<Map<String, dynamic>>> _getUserJobs(String userId) async {
   final snapshot = await _firestore
       .collection('jobs')
       .where('userId', isEqualTo: userId)
       .limit(5)
       .get();
   return snapshot.docs.map((doc) => doc.data()).toList();
 }

 static Future<int> _getReviewsCount(String userId) async {
   final snapshot = await _firestore
       .collection('reviews')
       .where('userId', isEqualTo: userId)
       .count()
       .get();
   return snapshot.count;
 }

 static Map<String, dynamic> _filterMobileUpdates(Map<String, dynamic>
updates) {
   const allowedFields = ['displayName', 'photoUrl', 'location', 'bio'];
   return {
     for (var entry in updates.entries)
       if (allowedFields.contains(entry.key)) entry.key: entry.value
   };
 }
}

extension ProfileExtensions on UsersRecord {
 Future<Map<String, dynamic>> getCompleteProfile() async {
   return ProfileService.getCompleteProfile(reference.id);
 }

 Future<void> updateProfile({
   required Map<String, dynamic> updates,
   bool isMobile = false,
 }) async {
   await ProfileService.updateProfileFields(
     userId: reference.id,
     updates: updates,
     isMobile: isMobile,
   );
   currentUserDocument = await UsersRecord.getDocumentOnce(reference);
 }

 Stream<List<Map<String, dynamic>>> streamReviews({int limit = 5}) {
   return FirebaseFirestore.instance
       .collection('reviews')
```

```
      .where('userId', isEqualTo: reference.id)
      .limit(limit)
      .snapshots()
      .map((snapshot) => snapshot.docs.map((doc) => doc.data()).toList());
 }
}

Future<void> initializeUserProfile(User user, {bool isMobile = false}) async {
 try {
   final exists = await UsersRecord.collection.doc(user.uid).get().then((
       doc) => doc.exists);
   if (exists) return;

   await UsersRecord.collection.doc(user.uid).set({
     'profile': {
       'bio': 'Tell us about yourself...',
       'skills': [],
       'rating': 0,
       'verified': false,
     },
     'settings': {
       'darkMode': false,
       'notifications': true,
       'language': 'en',
     },
     'createdAt': FieldValue.serverTimestamp(),
     'deviceType': isMobile ? 'mobile' : 'desktop',
   }, SetOptions(merge: true));
 } catch (e) {
   print('Error initializing user profile: $e');
 }
}
class ChatService {
 static final FirebaseFirestore _firestore = FirebaseFirestore.instance;

 static Future<void> sendMessage({
   required String chatId,
   required String senderId,
   required String message,
   required bool isProvider,
 }) async {
   await
_firestore.collection('chats').doc(chatId).collection('messages').add({
     'senderId': senderId,
     'message': message,
     'timestamp': FieldValue.serverTimestamp(),
     'isProvider': isProvider,
   });
```

```
  await _firestore.collection('chats').doc(chatId).update({
    'lastMessage': message,
    'lastMessageTime': FieldValue.serverTimestamp(),
  });
 }

 static Stream<List<Map<String, dynamic>>> getChatMessages(String chatId) {
   return _firestore
       .collection('chats')
       .doc(chatId)
       .collection('messages')
       .orderBy('timestamp', descending: true)
       .snapshots()
       .map((snapshot) => snapshot.docs.map((doc) => doc.data()).toList());
 }

 static Stream<List<Map<String, dynamic>>> getUserChats(String userId) {
   return _firestore
       .collection('chats')
       .where('participants', arrayContains: userId)
       .orderBy('lastMessageTime', descending: true)
       .snapshots()
       .map((snapshot) => snapshot.docs.map((doc) {
     final data = doc.data();
     return {
       'chatId': doc.id,
       ...data,
       'otherUser': data['participants'].firstWhere(
             (id) => id != userId,
         orElse: () => '',
       ),
     };
   }).toList());
 }

 static Future<Map<String, dynamic>> getChatJobDetails(String jobId) async {
   final doc = await _firestore.collection('jobs').doc(jobId).get();
   return doc.exists ? doc.data() ?? {} : {};
 }
}

extension ChatExtensions on UsersRecord {
 Future<void> sendChatMessage({
   required String chatId,
   required String message,
   required bool isProvider,
 }) async {
   await ChatService.sendMessage(
     chatId: chatId,
```

```
      senderId: reference.id,
      message: message,
      isProvider: isProvider,
    );
  }

  Stream<List<Map<String, dynamic>>> getChatsStream() {
    return ChatService.getUserChats(reference.id);
  }

  Stream<List<Map<String, dynamic>>> getChatMessagesStream(String chatId) {
    return ChatService.getChatMessages(chatId);
  }
}

class ChatState extends ChangeNotifier {
  bool _showJobPanel = false;
  Map<String, dynamic> _currentJobDetails = {};

  bool get showJobPanel => _showJobPanel;
  Map<String, dynamic> get currentJobDetails => _currentJobDetails;

  void toggleJobPanel() {
    _showJobPanel = !_showJobPanel;
    notifyListeners();
  }

  void setJobDetails(Map<String, dynamic> details) {
    _currentJobDetails = details;
    notifyListeners();
  }

  Future<void> loadJobDetails(String jobId) async {
    _currentJobDetails = await ChatService.getChatJobDetails(jobId);
    notifyListeners();
  }
}

class JobDetailsPanel extends StatelessWidget {
  final Map<String, dynamic> jobDetails;

  const JobDetailsPanel({super.key, required this.jobDetails});

  @override
  Widget build(BuildContext context) {
    return Container(
      width: 220,
      padding: const EdgeInsets.only(top: 45),
      child: SingleChildScrollView(
```

```
      child: Column(
        children: [
          Text(
            'Orders with You',
            style: FlutterFlowTheme.of(context).bodyMedium.override(
              fontSize: 22,
              fontWeight: FontWeight.w600,
            ),
          ),
          Padding(
            padding: const EdgeInsets.only(top: 5, bottom: 15),
            child: Text(
              'Total 05',
              style: FlutterFlowTheme.of(context).bodyMedium.override(
                color: FlutterFlowTheme.of(context).secondaryText,
                fontSize: 22,
              ),
            ),
          ),
          Text(
            'Current Order',
            style: FlutterFlowTheme.of(context).bodyMedium.override(
              fontSize: 22,
              fontWeight: FontWeight.w600,
            ),
          ),
          const SizedBox(height: 10),
          ClipRRect(
            borderRadius: BorderRadius.circular(8),
            child: Image.network(
              jobDetails['imageUrl'] ?? 'https://picsum.photos/seed/773/600',
              width: 200,
              height: 150,
              fit: BoxFit.cover,
            ),
          ),
          Padding(
            padding: const EdgeInsets.only(top: 20),
            child: Container(
              width: 200,
              decoration: BoxDecoration(
                color: FlutterFlowTheme.of(context).primaryBackground,
              ),
              child: Padding(
                padding: const EdgeInsets.all(10),
                child: Column(
                  crossAxisAlignment: CrossAxisAlignment.start,
                  children: [
```

```
                    _buildDetailRow('Budget', '\$${jobDetails['budget'] ??
'40'}'),
                    const SizedBox(height: 12),
                    _buildDetailRow('Duration', '${jobDetails['duration'] ??
'5'} hours'),
                    const SizedBox(height: 12),
                    Text(
                      'Job Status :',
                      style:
FlutterFlowTheme.of(context).bodyMedium.override(
                        fontSize: 20,
                        fontWeight: FontWeight.w500,
                      ),
                    ),
                    Align(
                      alignment: Alignment.centerRight,
                      child: Text(
                        jobDetails['status'] ?? 'In Progress',
                        style:
FlutterFlowTheme.of(context).bodyMedium.override(
                          color: Colors.green,
                          fontSize: 20,
                          fontWeight: FontWeight.w500,
                        ),
                      ),
                    ),
                  ],
                ),
              ),
            ),
          ),
          Padding(
            padding: const EdgeInsets.only(top: 30),
            child: FFButtonWidget(
              onPressed: () => print('Mark as Complete'),
              text: 'Mark as Complete',
              options: FFButtonOptions(
                width: 200,
                height: 54,
                color: const Color(0xFFE7E9EE),
                textStyle: FlutterFlowTheme.of(context).titleSmall.override(
                  color: const Color(0xFF203B7E),
                  fontSize: 20,
                  fontWeight: FontWeight.w600,
                ),
                borderRadius: BorderRadius.circular(16),
              ),
            ),
          ),
```

```
        ],
      ),
    ),
  );
}

Widget _buildDetailRow(String label, String value) {
  return Row(
    mainAxisAlignment: MainAxisAlignment.spaceBetween,
    children: [
      Text(
        label,
        style: FlutterFlowTheme.of(context).bodyMedium.override(
          fontSize: 20,
          fontWeight: FontWeight.w500,
        ),
      ),
      Text(
        value,
        style: FlutterFlowTheme.of(context).bodyMedium.override(
          fontSize: 20,
          fontWeight: FontWeight.w500,
        ),
      ),
    ],
  );
}
}
```