## main

import 'package:algorthimi/view/foot\_step\_counter.dart';  
import 'package:algorthimi/view/otp\_screen.dart';  
import 'package:flutter/material.dart';  
import 'package:get/get.dart';  
import 'package:google\_fonts/google\_fonts.dart';  
import 'package:permission\_handler/permission\_handler.dart';  
import 'package:responsive\_sizer/responsive\_sizer.dart';  
  
import 'controller/geo\_location\_controller.dart';  
import 'view/SplashLanguageScreens/splash\_screen.dart';  
import 'view/languages.dart';  
import 'package:firebase\_core/firebase\_core.dart';  
  
// void main() {  
// runApp(const MyApp());  
// }  
  
void main() async {  
 WidgetsFlutterBinding.ensureInitialized();  
 await Firebase.initializeApp();  
 runApp(MyApp());  
}  
  
class MyApp extends StatefulWidget {  
 const MyApp({Key? key}) : super(key: key);  
  
 @override  
 State<MyApp> createState() => \_MyAppState();  
}  
  
class \_MyAppState extends State<MyApp> {  
 LocationController locationController =  
 Get.put(LocationController(), tag: 'locationController');  
  
 @override  
 void initState() {  
 super.initState();  
 \_requestPedometerPermission();  
 }  
  
 Future<void> \_requestPedometerPermission() async {  
 final PermissionStatus status = await Permission.activityRecognition.status;  
 if (!status.isGranted) {  
 final PermissionStatus result = await Permission.activityRecognition.request();  
 if (!result.isGranted) {  
 // Handle denial of permission  
 print('Permission not granted');  
 }  
 }  
 }  
  
 @override  
 Widget build(BuildContext context) {  
 return ResponsiveSizer(  
 builder: (context, orientation, screenType) {  
 return GetMaterialApp(  
 debugShowCheckedModeBanner: false,  
 translations: Language(),  
 locale: const Locale('en', 'US'),  
 fallbackLocale: const Locale('en', 'US'),  
 theme: ThemeData(  
 fontFamily: GoogleFonts.urbanist().toString(),  
 useMaterial3: true,  
 colorScheme: const ColorScheme.light(background: Colors.white),  
 ),  
 home: const SplashScreen(),  
 // home: const FootStepCounter(),  
 );  
 },  
 );  
 }  
}

# controller

## geo\_location\_controller

import 'dart:developer';  
  
import 'package:geocoding/geocoding.dart';  
import 'package:geolocator/geolocator.dart';  
import 'package:get/get.dart';  
  
class LocationController extends GetxController {  
 var lat = 0.0.obs;  
 var lng = 0.0.obs;  
 dynamic currentPosition;  
 var currentAddress = ''.obs;  
  
 Future<Position> \_getGeoLocationPosition() async {  
 bool serviceEnabled;  
 LocationPermission permission;  
 // Test if location services are enabled.  
 serviceEnabled = await Geolocator.isLocationServiceEnabled();  
 if (!serviceEnabled) {  
 // Location services are not enabled don't continue  
 // accessing the position and request users of the  
 // App to enable the location services.  
 await Geolocator.openLocationSettings();  
 return Future.error('Location services are disabled.');  
 }  
 permission = await Geolocator.checkPermission();  
 if (permission == LocationPermission.denied) {  
 permission = await Geolocator.requestPermission();  
 if (permission == LocationPermission.denied) {  
 return Future.error('Location permissions are denied');  
 }  
 }  
 if (permission == LocationPermission.deniedForever) {  
 // Permissions are denied forever, handle appropriately.  
 return Future.error(  
 'Location permissions are permanently denied, we cannot request permissions.');  
 }  
 // When we reach here, permissions are granted and we can  
 // continue accessing the position of the device.  
 return await Geolocator.getCurrentPosition(  
 desiredAccuracy: LocationAccuracy.high);  
 }  
  
 // get address  
 Future<void> getAddressFromLatLong(Position position) async {  
 List<Placemark> placeMarks =  
 await placemarkFromCoordinates(position.latitude, position.longitude);  
 lat.value = position.latitude;  
 lng.value = position.longitude;  
  
 Placemark place = placeMarks[0];  
 currentAddress.value =  
 '${place.street}, ${place.subLocality}, ${place.locality}';  
  
 // log('Latitude: ${lat.value}');  
 // log('Longitude: ${lng.value}');  
 // log('place-marks: $placeMarks');  
 // log('currentAddress: ${currentAddress.value}');  
 }  
  
 // get address  
 getAddress() async {  
 Position position = await \_getGeoLocationPosition();  
 await getAddressFromLatLong(position);  
 }  
  
 getPosition() async {  
 Position position = await \_getGeoLocationPosition();  
 currentPosition = position;  
 log('here is');  
 await getAddressFromLatLong(currentPosition);  
 log("Position: $currentPosition");  
 }  
  
 @override  
 void onInit() {  
 // TODO: implement onInit  
 super.onInit();  
 getAddress();  
 }  
}

# utils

## app\_color

//colors  
const int kDarkGreyColor = 0xFF575757;  
const int kGreyColor = 0xFF4a4a4a;  
const int kLightGreyColor = 0xFF979595;  
const int kGradientGreyColor = 0xFF8F939E;  
const int kDeepDarkGreenColor = 0xff014531;  
const int kDarkGreenColor = 0xFF1c8164;  
const int kGreenShadowColor = 0xff6ddfbf;  
const int kSeaGreenColor = 0xFF00BCD4;  
const int kLightSeaGreenColor = 0xFFE6C0F8FF;  
const int kShadowColor = 0xFFe9e9e9;  
const int purple = 0xFF4111ab;  
const int pink = 0xFFee0060;

# utils

## color\_data

import 'dart:ui';  
  
Color accentColor = "#3E13AA".toColor();  
Color pinkAppColor = "#F10062".toColor();  
Color borderColor = "#E8ECF4".toColor();  
Color blackColor = "#000000".toColor();  
Color backgroundColor = "#FFFFFF".toColor();  
Color textBlackColor = "#1E1E1E".toColor();  
Color textWhiteColor = "#FFFFFF".toColor();  
Color subTextColor = "#999EA1".toColor();  
Color skyBlueTextColor = "#35C2C1".toColor();  
Color hintTextColor = "#1F1F1F6E".toColor();  
Color forgotPasswordColor = "#FB344F".toColor();  
Color skipBDColor = "#F5F5F5".toColor();  
Color bottomBarIconColor = "#6D6E71".toColor();  
Color inactiveSliderColor = "#D9D9D9".toColor();  
Color feePayCardColor = "#C9DEFA".toColor();  
Color backCardColor = "#EAF2FF".toColor();  
Color redColor = "#F20707".toColor();  
Color greenColor = "#0A9555".toColor();  
Color orangeColor = "#F68C25".toColor();  
Color lightGreenColor = "#60BF84".toColor();  
Color blue = "#26E0E7".toColor();  
Color lightblue = "#CFEAFF".toColor();  
Color lightpink = "#FFACD4".toColor();  
  
extension ColorExtension on String {  
 toColor() {  
 var hexColor = replaceAll("#", "");  
 if (hexColor.length == 6) {  
 hexColor = "FF$hexColor";  
 }  
 if (hexColor.length == 8) {  
 return Color(int.parse("0x$hexColor"));  
 }  
 }  
}

# utils

## constant

import 'package:flutter/material.dart';  
import 'package:flutter/services.dart';  
import 'package:get/get.dart';  
  
class Constant {  
 static String assetImagePath = "assets/images/";  
 static String svgImagePath = "assets/svg/";  
 static bool isDriverApp = false;  
 static const String fontsFamily = "Urbanist";  
  
 static backToPrev(BuildContext context) {  
 Get.back();  
 }  
  
 static getCurrency(BuildContext context) {  
 return "ETH";  
 }  
  
 static sendToNext(BuildContext context, String route, {Object? arguments}) {  
 if (arguments != null) {  
 Get.toNamed(route, arguments: arguments);  
 } else {  
 Get.toNamed(route);  
 }  
 }  
  
 static double getToolbarHeight(BuildContext context) {  
 return MediaQuery.of(context).padding.top + kToolbarHeight;  
 }  
  
 static double getToolbarTopHeight(BuildContext context) {  
 return MediaQuery.of(context).padding.top;  
 }  
  
 static sendToScreen(Widget widget, BuildContext context) {  
 Get.to(widget);  
 }  
  
 static backToFinish(BuildContext context) {  
 Get.back();  
 }  
  
 static closeApp() {  
 Future.delayed(const Duration(milliseconds: 1000), () {  
 SystemChannels.platform.invokeMethod('SystemNavigator.pop');  
 });  
 }  
}

# utils

## data

import 'package:get/get.dart';  
import 'package:google\_maps\_flutter/google\_maps\_flutter.dart';  
  
class DataFile {  
 static List<LanguageModel> languageList = [  
 LanguageModel(  
 "English",  
 ),  
 LanguageModel(  
 "عربي",  
 ),  
 LanguageModel(  
 "Français",  
 ),  
 LanguageModel(  
 "Deutsch",  
 ),  
 LanguageModel(  
 "اردو",  
 ),  
 LanguageModel(  
 "हिंदी",  
 ),  
 LanguageModel(  
 "മലയാളം",  
 ),  
 LanguageModel(  
 "বাংলা",  
 ),  
 LanguageModel(  
 "नेपाली",  
 ),  
 ];  
 static List<QuestionsModel> questionsModel = [  
 QuestionsModel(  
 "1",  
 "1",  
 "Does the location have a POS system?",  
 ),  
 QuestionsModel(  
 "2",  
 "1",  
 "Total Numbers of cash counters?",  
 ),  
 QuestionsModel(  
 "3",  
 "2",  
 "Do they sale tobacco?",  
 ),  
 QuestionsModel(  
 "4",  
 "3",  
 "Attach image of tobacco shelf.",  
 ),  
 QuestionsModel(  
 "5",  
 "4",  
 "Ask shopkeeper top selling brands?",  
 ),  
 ];  
 static List<CitiesLatLongModel> citiesLatLongModel = [  
 CitiesLatLongModel(  
 BitmapDescriptor.hueCyan, "gulberg", "Gulberg", 31.5102, 74.3441),  
 CitiesLatLongModel(  
 BitmapDescriptor.hueBlue, "model-town", "Model Town", 31.4805, 74.3239),  
 CitiesLatLongModel(  
 BitmapDescriptor.hueAzure, "township", "Township", 31.4475, 74.3081),  
 ];  
  
 static List<BankMethodModel> bankMethodModel = [  
 BankMethodModel("First Abu Dhabi Bank"),  
 BankMethodModel("Dubai Islamic Bank"),  
 BankMethodModel("Etisalat Wallet"),  
 BankMethodModel("Magnati"),  
 ];  
  
}  
  
///// Models  
  
class LanguageModel {  
 String? name;  
  
 LanguageModel(this.name);  
}  
  
/// Bank Method  
class BankMethodModel {  
 String? nameBank;  
  
 BankMethodModel(this.nameBank);  
}  
  
class CitiesLatLongModel {  
 double markerColor;  
 String? markerID;  
 String? cityName;  
 double cityLat;  
 double cityLong;  
  
 CitiesLatLongModel(this.markerColor, this.markerID, this.cityName,  
 this.cityLat, this.cityLong);  
}  
  
class QuestionsModel {  
 String? questionNo;  
 String? questionType;  
 String? questionTitle;  
  
 QuestionsModel(this.questionNo, this.questionType, this.questionTitle);  
}

# view

## custom\_dialog

import 'package:flutter/material.dart';  
import 'package:algorthimi/utils/app\_color.dart';  
import 'package:cnic\_scanner/cnic\_scanner.dart';  
  
class CustomDialogBox extends StatefulWidget {  
 Function onCameraBTNPressed, onGalleryBTNPressed;  
  
 CustomDialogBox(  
 {required this.onCameraBTNPressed, required this.onGalleryBTNPressed});  
  
 @override  
 \_CustomDialogBoxState createState() => \_CustomDialogBoxState();  
}  
  
class \_CustomDialogBoxState extends State<CustomDialogBox> {  
 @override  
 Widget build(BuildContext context) {  
 return Dialog(  
 shape: RoundedRectangleBorder(  
 borderRadius: BorderRadius.circular(20),  
 ),  
 elevation: 0,  
 backgroundColor: Colors.transparent,  
 child: contentBox(context),  
 );  
 }  
  
 contentBox(context) {  
 return Stack(  
 children: <Widget>[  
 Container(  
 padding: EdgeInsets.only(left: 20, top: 10, right: 20, bottom: 10),  
 margin: EdgeInsets.only(top: 45),  
 decoration: BoxDecoration(  
 shape: BoxShape.rectangle,  
 color: Colors.white,  
 borderRadius: BorderRadius.circular(20),  
 boxShadow: [  
 BoxShadow(  
 color: Colors.black, offset: Offset(0, 10), blurRadius: 10),  
 ]),  
 child: Column(  
 mainAxisSize: MainAxisSize.min,  
 children: <Widget>[  
 Text(  
 'Cnic Scan',  
 style: TextStyle(fontSize: 25, fontWeight: FontWeight.w600),  
 ),  
 SizedBox(  
 height: 15,  
 ),  
 Text(  
 'Please open your camera',  
 style: TextStyle(fontSize: 15),  
 textAlign: TextAlign.center,  
 ),  
 SizedBox(  
 height: 22,  
 ),  
 Row(  
 mainAxisAlignment: MainAxisAlignment.spaceAround,  
 children: [  
 Focus(  
 onFocusChange: (hasFocus) {  
 },  
 child: ElevatedButton(  
 style: ElevatedButton.styleFrom(  
 elevation: 5,  
 shape: RoundedRectangleBorder(  
 borderRadius: BorderRadius.circular(10.0),  
 ),  
 textStyle: TextStyle(color: Colors.white),  
 padding: EdgeInsets.all(0.0),  
 ),  
 onPressed: () {  
 Navigator.pop(context);  
 widget.onCameraBTNPressed();  
 },  
 child: Column(  
 children: [  
 Container(  
 alignment: Alignment.center,  
 width: 150,  
 decoration: BoxDecoration(  
 color: Color(pink),  
 borderRadius:  
 BorderRadius.all(Radius.circular(10.0)),  
 ),  
 padding: const EdgeInsets.all(12.0),  
 child: Text('Open Camera',  
 style: TextStyle(  
 fontSize: 18, color: Colors.white)),  
 ),  
 ],  
 ),  
 ),  
 )  
  
 // ElevatedButton(  
 // style: ElevatedButton.styleFrom(  
 // elevation: 5,  
 // shape: RoundedRectangleBorder(borderRadius: BorderRadius.circular(10.0)),  
 // textStyle: TextStyle(color: Colors.white),  
 // padding: EdgeInsets.all(0.0),  
 // ),  
 // onPressed: () {{Navigator.pop(context);widget.onCameraBTNPressed();}  
 // },  
 // child: Column(  
 // children: [  
 // Container(  
 // alignment: Alignment.center,  
 // width: 150,  
 // decoration: BoxDecoration(  
 // color: Color(pink),  
 // borderRadius: BorderRadius.all(Radius.circular(10.0)),),  
 // padding: const EdgeInsets.all(12.0),  
 // child: Text('Open Camera', style: TextStyle(fontSize: 18, color: Colors.white)),  
 // ),  
 // ],  
 // ),  
 // ),  
  
 // ElevatedButton(onPressed: () {Navigator.pop(context);widget.onCameraBTNPressed();},  
 //  
 // child: Text('Open Camera', style: TextStyle(fontSize: 18, color: Color(purple)),)  
 // ),  
  
 // TextButton(onPressed: () {Navigator.pop(context);widget.onGalleryBTNPressed();},  
 //  
 // child: Text('GALLERY', style: TextStyle(fontSize: 18, color: Color(purple)),)),  
 ],  
 ),  
 ],  
 ),  
 ),  
  
 // Positioned(  
 // left: 20,  
 // right: 20,  
 // child: CircleAvatar(  
 // backgroundColor: Colors.transparent,  
 // radius: 45,  
 // child: ClipRRect(  
 // borderRadius: BorderRadius.all(Radius.circular(45)),  
 // child: Image.asset("assets/images/person\_icon.png")),  
 // ),  
 // ),  
 ],  
 );  
 }  
}

# view

## foot\_step\_counter

import 'package:algorthimi/view/AuthScreens/welldone\_screen.dart';  
import 'package:flutter/material.dart';  
import 'package:permission\_handler/permission\_handler.dart';  
import 'package:get/get.dart';  
import 'package:get/get\_core/src/get\_main.dart';  
import 'package:algorthimi/view/BottomNavBar/bottom\_nav\_bar\_freelancer\_screen.dart';  
import 'package:responsive\_sizer/responsive\_sizer.dart';  
  
import '../../utils/color\_data.dart';  
import 'dart:async';  
import 'package:pedometer/pedometer.dart';  
import 'package:intl/intl.dart';  
import '../utils/app\_color.dart';  
import 'Widgets/widget\_utils.dart';  
  
class FootStepCounter extends StatefulWidget {  
 const FootStepCounter({super.key});  
  
 @override  
 State<FootStepCounter> createState() => \_FootStepCounterState();  
}  
  
class \_FootStepCounterState extends State<FootStepCounter> {  
 late Stream<StepCount> \_stepCountStream;  
 int \_totalSteps = 0;  
 late int minus = 0;  
 late int plus = 0;  
 static late int cal;  
 bool \_isCountingSteps = false;  
 final Stopwatch \_stopwatch = Stopwatch();  
 late Timer \_timer;  
 late DateTime \_startDate;  
 bool \_startButtonClicked = false;  
  
 // Conversion factor: 1 meter^2 = 10.7639 square feet  
 final double squareFeetPerSquareMeter = 2.50;  
  
 // Conversion factor: 1 square meter = 0.005 marlas  
 final double marlasPerSquareMeter = 0.005;  
  
 // Assuming an average stride length of 0.75 meters  
 final double averageStrideLengthMeters = 0.75;  
  
 @override  
 void initState() {  
 super.initState();  
 \_startDate = DateTime.now();  
 \_initPedometer();  
 }  
  
  
  
 void \_initPedometer() {  
 \_stepCountStream = Pedometer.stepCountStream;  
 bool isCalculated = false; // Flag to track if cal has been assigned  
 \_stepCountStream.listen((StepCount event) {  
 if (\_isCountingSteps) {  
 setState(() {  
 if (\_startDate.day != DateTime.now().day) {  
 \_resetSteps();  
 }  
 \_totalSteps = event.steps;  
 if (!isCalculated) {  
 cal = \_totalSteps;  
 isCalculated = true; // Set the flag to true after assigning cal  
 }  
 minus = \_totalSteps - cal;  
 // plus = ((\_totalSteps - cal) - minus);  
  
 });  
 }  
 });  
 }  
  
 void \_inPedometer() {  
 \_stepCountStream = Pedometer.stepCountStream;  
 bool isCalculated = false; // Flag to track if cal has been assigned  
 \_stepCountStream.listen((StepCount event) {  
 if (\_isCountingSteps) {  
 setState(() {  
 if (\_startDate.day != DateTime.now().day) {  
 \_resetSteps();  
 }  
 \_totalSteps = event.steps;  
 if (!isCalculated) {  
 cal = \_totalSteps;  
 isCalculated = true; // Set the flag to true after assigning cal  
 }  
 plus = \_totalSteps - cal;  
  
 });  
 }  
 }  
 );  
 }  
  
 void \_resetSteps() {  
 \_startDate = DateTime.now();  
 \_totalSteps = 0;  
 // minus = \_totalSteps - 6774; // Update minus whenever \_totalSteps changes  
 }  
  
 void \_startCountingSteps() {  
 setState(() {  
 \_isCountingSteps = true;  
 \_stopwatch.start();  
 \_startTimer();  
 // Set the flag to true when Start button is clicked  
 \_startButtonClicked = false;  
 });  
 }  
  
 void \_stopCountingSteps() {  
 setState(() {  
 \_isCountingSteps = false;  
 \_stopwatch.stop();  
 \_stopTimer();  
 });  
 }  
  
 void \_resetTime() {  
 setState(() {  
 \_stopwatch.reset();  
 });  
 }  
  
 void \_startTimer() {  
 \_timer = Timer.periodic(const Duration(seconds: 1), (timer) {  
 setState(() {});  
 });  
 }  
  
 void \_stopTimer() {  
 \_timer.cancel();  
 }  
  
 String \_formatElapsedTime(Duration duration) {  
 final minutes = duration.inMinutes;  
 final seconds = duration.inSeconds % 60;  
 return '$minutes:${seconds.toString().padLeft(2, '0')}';  
 }  
  
 double \_calculateDistanceInSquareMeters(int steps) {  
 return steps \* averageStrideLengthMeters;  
 }  
  
 double \_convertSquareMetersToSquareFeet(double squareMeters) {  
 return squareMeters \* squareFeetPerSquareMeter;  
 }  
  
 double \_convertSquareMetersToMarlas(double squareMeters) {  
 return squareMeters \* marlasPerSquareMeter;  
 }  
  
 //////////////////////////////////////////////////////////////////////  
  
 int \_clickCount = 0;  
 List<String> buttonTexts = [  
 'Click here',  
 'Stop',  
 'Start walking left or right',  
 'Stop'  
 ];  
 bool saveButtonEnabled = false; // Track if Save button should be enabled  
  
 void \_changeText() {  
 setState(() {  
 if (\_clickCount < 4) {  
 \_clickCount++;  
 }  
 if (\_clickCount == 1 || \_clickCount == 3) {  
 // Start counting steps  
 \_startCountingSteps();  
 } else if (\_clickCount == 2 || \_clickCount == 4) {  
 // Stop counting steps  
 \_stopCountingSteps();  
 }  
 if (\_clickCount == 4) {  
 saveButtonEnabled = true; // Enable Save button after clicking 4 times  
 }  
  
 // Update the footsteps count and calculate distance  
 if (\_clickCount == 3) {  
 // Calculate new steps when user starts walking left or right  
 \_inPedometer();  
 }  
 });  
 }  
  
  
 @override  
 Widget build(BuildContext context) {  
 double distanceInSquareMeters = \_calculateDistanceInSquareMeters(minus);  
 double distanceInSquareFeet = \_convertSquareMetersToSquareFeet(distanceInSquareMeters);  
  
 double distanceInSquareMeters2 = \_calculateDistanceInSquareMeters(plus);  
 double distanceInSquareFeet2 = \_convertSquareMetersToSquareFeet(distanceInSquareMeters2);  
  
 return Scaffold(  
 appBar: AppBar(  
 centerTitle: true,  
 title: const Text(  
 'Area measurement of store',  
 style: TextStyle(fontWeight: FontWeight.w600, color: Colors.white),  
 ),  
 backgroundColor: const Color(purple),  
 iconTheme: const IconThemeData(color: Colors.white),  
 ),  
  
 body: SingleChildScrollView(  
 child: Column(  
 mainAxisAlignment: MainAxisAlignment.start,  
 children: <Widget>[  
 const SizedBox(  
 height: 40,  
 ),  
 Row(  
 mainAxisAlignment: MainAxisAlignment.center,  
 children: [  
 const Text(  
 'timer : ',  
 style: TextStyle(fontSize: 24, fontWeight: FontWeight.bold),  
 ),  
 Text(  
 \_formatElapsedTime(\_stopwatch.elapsed),  
 style: const TextStyle(  
 fontSize: 24,  
 ),  
 ),  
 ],  
 ),  
  
 const SizedBox(  
 height: 10,  
 ),  
  
  
 Row(  
 mainAxisAlignment: MainAxisAlignment.spaceAround,  
 children: [  
  
 Container(  
 width: 150,  
 height: 100,  
 decoration: BoxDecoration(  
 borderRadius: BorderRadius.circular(20.0),  
 border: Border.all(  
 color: Colors.black,  
 // You can specify any color you want here  
 width: 2.0, // You can adjust the width of the border  
 ),  
 ),  
 child: Column(  
 mainAxisAlignment: MainAxisAlignment.center,  
 children: [  
 Container(  
 child: getCustomTextW6S20(  
 text: 'Side 1'.tr, color: Colors.black),  
 ),  
 const Divider(  
 thickness: 2,  
 ),  
 Row(  
 mainAxisAlignment: MainAxisAlignment.center,  
 children: [  
 getCustomTextW6S15(  
 text: 'Steps : '.tr, color: Colors.black),  
 getCustomTextW6S15(  
 text: '$minus', color: Colors.black),  
 ],  
 ),  
 Row(  
 mainAxisAlignment: MainAxisAlignment.center,  
 children: [  
 getCustomTextW6S15(  
 text: 'Area : '.tr, color: Colors.black),  
 getCustomTextW6S15(  
 text: '${distanceInSquareFeet.toStringAsFixed(2)}'  
 .tr,  
 color: Colors.black),  
 ],  
 ),  
 ],  
 ),  
 ),  
  
  
 Container(  
 width: 150,  
 height: 100,  
 decoration: BoxDecoration(  
 borderRadius: BorderRadius.circular(20.0),  
 border: Border.all(  
 color: Colors.black,  
 // You can specify any color you want here  
 width: 2.0, // You can adjust the width of the border  
 ),  
 ),  
 child: Column(  
 mainAxisAlignment: MainAxisAlignment.center,  
 children: [  
 getCustomTextW6S20(  
 text: 'Side 2'.tr, color: Colors.black),  
 const Divider(  
 thickness: 2,  
 ),  
 Row(  
 mainAxisAlignment: MainAxisAlignment.center,  
 children: [  
 getCustomTextW6S15(  
 text: 'Steps : '.tr, color: Colors.black),  
 getCustomTextW6S15(  
 text: '$plus'.tr, color: Colors.black),  
 ],  
 ),  
 Row(  
 mainAxisAlignment: MainAxisAlignment.center,  
 children: [  
 getCustomTextW6S15(  
 text: 'Area : '.tr, color: Colors.black),  
 getCustomTextW6S15(  
 text: '${distanceInSquareFeet2.toStringAsFixed(2)}'.tr, color: Colors.black),  
 ],  
 ),  
 ],  
 ),  
 ),  
 ],  
 ),  
  
  
 const SizedBox(  
 height: 30,  
 ),  
  
  
 Container(  
 height: 50,  
 width: 300,  
 decoration: BoxDecoration(  
 borderRadius: BorderRadius.circular(8.0),  
 border: Border.all(  
 color: Colors.black,  
 // You can specify any color you want here  
 width: 2.0, // You can adjust the width of the border  
 ),  
 ),  
 child: Center(  
 child: Row(  
 mainAxisAlignment: MainAxisAlignment.center,  
 children: [  
 getCustomTextW6S22(  
 text: 'Total area : '.tr, color: Colors.black),  
 getCustomTextW6S22(text: '${(distanceInSquareFeet \* distanceInSquareFeet2).toStringAsFixed(2)} sq ft'.tr, color: Colors.black),  
 ],  
 ),  
 )),  
  
  
  
 const SizedBox(  
 height: 50,  
 ),  
  
  
 Container(  
 width: 200,  
 height: 50,  
 child: GestureDetector(  
 onTap: () {  
 \_changeText();  
 // \_startCountingSteps;  
 },  
 child: fillColorButton(  
 color: pinkAppColor,  
 text: \_clickCount < 4 ? buttonTexts[\_clickCount] : 'Area calculate complete'.tr,  
 ),  
 ),  
 ),  
  
  
 const SizedBox(  
 height: 40,  
 ),  
  
  
 Row(  
 mainAxisAlignment: MainAxisAlignment.center,  
 children: [  
 // Text(  
 // '$\_totalSteps',  
 // style: const TextStyle(fontSize: 40, fontWeight: FontWeight.bold),  
 // ),  
 Container(  
 width: 110,  
 height: 40,  
 child: GestureDetector(  
 onTap: () {  
 // \_stopCountingSteps;  
 },  
 child: fillColorButton(  
 // color: pinkAppColor,  
 text: 'Start walking'.tr,  
 color: \_totalSteps > 0 ? const Color(purple): Colors.grey,  
 ),  
 ),  
 ),  
  
  
 const SizedBox(  
 width: 50,  
 ),  
  
  
 Container(  
 width: 110,  
 height: 40,  
 child: GestureDetector(  
 onTap: saveButtonEnabled  
 ? () {  
 \_startCountingSteps();  
 // Your onPressed functionality here  
 }  
 : null,  
 child: Container(  
 decoration: saveButtonEnabled  
 ? BoxDecoration(  
 color: const Color(purple),  
 borderRadius: BorderRadius.circular(  
 8.0), // Add border radius if needed  
 )  
 : BoxDecoration(  
 color: Colors.grey,  
 borderRadius: BorderRadius.circular(  
 8.0), // Add border radius if needed  
 ),  
 child: const Padding(  
 padding: EdgeInsets.all(10.0),  
 child: Center(  
 child: Text(  
 'Save',  
 style: TextStyle(  
 color: Colors.white,  
 // Change text color according to your design  
 fontSize:  
 16.0, // Change font size according to your design  
 ),  
 ),  
 ),  
 ),  
 ),  
 ),  
 ),  
 ],  
 ),  
 //  
  
 const SizedBox(  
 height: 135,  
 ),  
  
 Padding(  
 padding: const EdgeInsets.all(20.0),  
 child: GestureDetector(  
 onTap: () {  
 if (saveButtonEnabled) {  
 Get.to(() => WellDoneScreen());  
 // Get.offAll(() => const BottomNavBarFreelancerScreen());  
 } else {  
 Get.snackbar(  
 'Alert',  
 'Please calculate shop area and click save',  
 snackPosition: SnackPosition.TOP,  
 backgroundColor: const Color(pink),  
 colorText: Colors.white,  
 );  
 }  
 },  
 child: fillColorButton(  
 color: pinkAppColor,  
 text: 'Next'.tr,  
 ),  
 ),  
 )  
 ],  
 ),  
 ),  
 );  
 }  
}

# view

## id\_scan

import 'package:algorthimi/view/otp\_screen.dart';  
import 'package:flutter/material.dart';  
import 'package:get/get.dart';  
import 'package:image\_picker/image\_picker.dart';  
import 'package:cnic\_scanner/cnic\_scanner.dart';  
  
import 'Widgets/widget\_utils.dart';  
import 'custom\_dialog.dart';  
import 'package:algorthimi/view/AuthScreens/scan\_id\_freelancer\_screen.dart';  
import 'package:cnic\_scanner/model/cnic\_model.dart';  
import 'package:algorthimi/utils/app\_color.dart';  
  
class CnicScan extends StatelessWidget {  
 @override  
 Widget build(BuildContext context) {  
 return MaterialApp(  
 title: 'CNIC Scanner',  
 debugShowCheckedModeBanner: false,  
 theme: ThemeData(  
 primarySwatch: Colors.green,  
 ),  
 home: MyHomePage(),  
 );  
 }  
}  
  
class MyHomePage extends StatefulWidget {  
 @override  
 \_MyHomePageState createState() => \_MyHomePageState();  
}  
  
class \_MyHomePageState extends State<MyHomePage> {  
 TextEditingController nameTEController = TextEditingController(),  
 cnicTEController = TextEditingController(),  
 dobTEController = TextEditingController(),  
 doiTEController = TextEditingController(),  
 doeTEController = TextEditingController();  
  
 CnicModel \_cnicModel = CnicModel();  
 bool \_isCnicScanned = false;  
  
 Future<void> scanCnic(ImageSource imageSource) async {  
 CnicModel cnicModel =  
 await CnicScanner().scanImage(imageSource: imageSource);  
 if (cnicModel == null) return;  
 setState(() {  
 \_cnicModel = cnicModel;  
 nameTEController.text = \_cnicModel.cnicHolderName;  
 cnicTEController.text = \_cnicModel.cnicNumber;  
 dobTEController.text = \_cnicModel.cnicHolderDateOfBirth;  
 doiTEController.text = \_cnicModel.cnicIssueDate;  
 doeTEController.text = \_cnicModel.cnicExpiryDate;  
 \_isCnicScanned = true; // Set flag to true when CNIC is scanned  
 });  
 }  
  
 @override  
 Widget build(BuildContext context) {  
 return Scaffold(  
 appBar: AppBar(  
 centerTitle: true,  
 title: Row(  
 mainAxisAlignment: MainAxisAlignment.spaceBetween,  
 children: [  
 GestureDetector(  
 onTap: () {  
 Get.back();// Navigate back when the icon is tapped  
 },  
 child: Icon(Icons.arrow\_back\_ios\_new, color: Colors.white),  
 ),  
 Padding(  
 padding: const EdgeInsets.fromLTRB(0, 0, 110, 0),  
 child: Text(  
 'Scan Your ID',  
 style: TextStyle(fontWeight: FontWeight.w600, color: Colors.white),  
 ),  
 ),  
 ],  
 ),  
  
 backgroundColor: Color(purple),  
 ),  
  
 body: Container(  
 margin: const EdgeInsets.only(  
 left: 20, right: 20, top: 50, bottom: 25),  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.start,  
 children: [  
 SizedBox(  
 height: 18,  
 ),  
 Text(  
 'Enter ID Card Details...',  
 style: TextStyle(  
 color: Color(kDarkGreyColor),  
 fontSize: 23.0,  
 fontWeight: FontWeight.bold),  
 ),  
 SizedBox(  
 height: 35,  
 ),  
 Expanded(  
 child: ListView(  
 padding: const EdgeInsets.all(0),  
 shrinkWrap: true,  
 children: [  
 \_dataField(  
 text: 'Name',  
 textEditingController: nameTEController),  
 \_cnicField(  
 textEditingController: cnicTEController),  
 \_dataField(  
 text: 'Date of Birth',  
 textEditingController: dobTEController),  
 \_dataField(  
 text: 'Date of Card Issue',  
 textEditingController: doiTEController),  
 \_dataField(  
 text: 'Date of Card Expire',  
 textEditingController: doeTEController),  
 SizedBox(  
 height: 20,  
 ),  
 \_getScanCNICBtn(),  
 ],  
 ),  
 )  
 ],  
 ),  
 ),  
 );  
 }  
  
 Widget \_cnicField(  
 {required TextEditingController textEditingController}) {  
 return Card(  
 elevation: 7,  
 margin: const EdgeInsets.only(top: 2.0, bottom: 5.0),  
 child: Container(  
 margin: const EdgeInsets.only(  
 top: 2.0, bottom: 1.0, left: 0.0, right: 0.0),  
 child: IntrinsicHeight(  
 child: Row(  
 crossAxisAlignment: CrossAxisAlignment.stretch,  
 children: [  
 Container(  
 width: 3,  
 height: 45,  
 margin: const EdgeInsets.only(left: 3.0, right: 7.0),  
 ),  
 Expanded(  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.start,  
 children: [  
  
 Row(  
 mainAxisAlignment: MainAxisAlignment.start,  
 children: [  
 Padding(  
 padding: const EdgeInsets.fromLTRB(0, 10, 0, 0),  
 child: Icon(Icons.credit\_card\_outlined,  
 color: Color(purple),  
 size: 17,),  
 ),  
 Padding(  
 padding: const EdgeInsets.only(left: 15.0),  
 child: Text(  
 'CNIC Number',  
 style: TextStyle(  
 color: Color(purple),  
 fontSize: 13.0,  
 fontWeight: FontWeight.bold),  
 ),  
 ),  
 ],  
 ),  
 Row(  
 children: [  
 Expanded(  
 child: TextField(  
 controller: textEditingController,  
 decoration: InputDecoration(  
 hintText: '00000-0000000-0',  
 hintStyle:  
 TextStyle(color: Color(kLightGreyColor)),  
 border: InputBorder.none,  
 isDense: true,  
 contentPadding: EdgeInsets.only(left: 32.0),  
 ),  
 style: TextStyle(  
 color: Color(kDarkGreyColor),  
 fontWeight: FontWeight.bold),  
 textInputAction: TextInputAction.done,  
 keyboardType: TextInputType.number,  
 textAlign: TextAlign.left,  
 ),  
 )  
 ],  
 )  
 ],  
 )),  
 ],  
 ),  
 ),  
 ),  
 );  
 }  
  
 Widget \_dataField(  
 {required String text,  
 required TextEditingController textEditingController}) {  
 return Card(  
 shadowColor: Colors.green,  
 elevation: 5,  
 margin: const EdgeInsets.only(top: 10, bottom: 5,),  
 child: Row(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 mainAxisAlignment: MainAxisAlignment.spaceAround,  
 children: [  
 Padding(  
 padding: const EdgeInsets.only(left: 14),  
 child: Icon(  
 (text == "Name") ? Icons.person : Icons.date\_range,  
 color: Color(purple),  
 size: 17,  
 ),  
 ),  
 Expanded(  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.start,  
 mainAxisAlignment: MainAxisAlignment.spaceAround,  
 children: [  
 Padding(  
 padding:  
 const EdgeInsets.only(left: 15.0, top: 5, bottom: 3),  
 child: Text(text.toUpperCase(),  
 style: TextStyle(color: Color(purple), fontSize: 12, fontWeight: FontWeight.bold),),  
 ),  
 Padding(  
 padding: const EdgeInsets.only(left: 15.0, bottom: 5),  
 child: TextField(  
 controller: textEditingController,  
 decoration: InputDecoration(  
 hintText: (text == "Name") ? "User Name" : 'DD/MM/YYYY',  
 border: InputBorder.none,  
 isDense: true,  
 hintStyle: TextStyle(color: Color(kLightGreyColor), fontSize: 14, fontWeight: FontWeight.bold), contentPadding: EdgeInsets.all(0),  
 ),  
 style: TextStyle(color: Color(kDarkGreyColor), fontWeight: FontWeight.bold),  
 textInputAction: TextInputAction.done,  
 keyboardType: (text == "Name") ? TextInputType.text : TextInputType.number,  
 textAlign: TextAlign.left,  
 ),  
 ),  
 ],  
 ),  
 ),  
 ],  
 ));  
 }  
  
 Widget \_getScanCNICBtn() {  
 return Column(  
 children: [  
 ElevatedButton(  
 style: ElevatedButton.styleFrom(  
 elevation: 5,  
 shape: RoundedRectangleBorder(borderRadius: BorderRadius.circular(10.0)),  
 textStyle: TextStyle(color: Colors.white),  
 padding: EdgeInsets.all(0.0),  
 ),  
 onPressed: () {  
 showDialog(context: context, builder: (BuildContext context) {  
 return CustomDialogBox(onCameraBTNPressed: () {scanCnic(ImageSource.camera);},  
 onGalleryBTNPressed: () {scanCnic(ImageSource.gallery);});  
 });  
 },  
 child: Column(  
 children: [  
 Container(  
 alignment: Alignment.center,  
 width: 500,  
 decoration: BoxDecoration(  
 color: Color(pink),  
 borderRadius: BorderRadius.all(Radius.circular(10.0)),),  
 padding: const EdgeInsets.all(12.0),  
 child: Text('Scan CNIC', style: TextStyle(fontSize: 18, color: Colors.white)),  
 ),  
 ],  
 ),  
 ),  
 SizedBox(height: 20,),  
 GestureDetector(  
 onTap: () {  
 if (\_isCnicScanned) {  
 Get.to(() => const OtpScreen());  
 // Get.to(() => const ScanIdFreelancerScreen());  
 } else {  
 // Show a message that CNIC needs to be scanned first  
 ScaffoldMessenger.of(context).showSnackBar(  
 SnackBar(  
 duration: Duration(seconds: 5),  
 content: Text('Please scan your CNIC first.'),  
 ),  
 );  
 }  
 },  
 child: IgnorePointer(  
 ignoring: !\_isCnicScanned,  
 child: AnimatedOpacity(  
 duration: Duration(milliseconds: 200),  
 opacity: \_isCnicScanned ? 1.0 : 0.5,  
 child: fillColorButton(  
 color: Color(purple),  
 text: 'Next'.tr,  
 ),  
 ),  
 ),  
 ),  
 ],  
 );  
 }  
}

# view

## languages

import 'package:get/get\_navigation/src/root/internacionalization.dart';  
  
class Language extends Translations {  
 @override  
 Map<String, Map<String, String>> get keys => {  
 'en\_US': {  
 'Select your preferred language': 'Select your preferred language',  
 'Login': 'Login',  
 'Enter Your Email': 'Enter Your Email',  
 'Please enter a valid email': 'Please enter a valid email',  
 'Enter Your Password': 'Enter Your Password',  
 'Password not be empty!': 'Password not be empty!',  
 'Password must be greater than 5 character!':  
 'Password must be greater than 5 character!',  
 'Forgot Password?': 'Forgot Password?',  
 'Forgot Password': 'Forgot Password',  
 'Don’t have an account ? ': 'Don’t have an account ? ',  
 'Sign Up': 'Sign Up',  
 'Don’t worry! It happens. Please enter the email address associated with your account.':  
 'Don’t worry! It happens. Please enter the email address associated with your account.',  
 'Submit': 'Submit',  
 'Reset Password': 'Reset Password',  
 'Your new password must be different from previous used passwords.':  
 'Your new password must be different from previous used passwords.',  
 'Password': 'Password',  
 'Confirm Password': 'Confirm Password',  
 'Select Role': 'Select Role',  
 'Freelancers': 'Freelancers',  
 'Shop Owner': 'Shop Owner',  
 'Already have an account ? ': 'Already have an account ? ',  
 'Upload Selfie': 'Upload Selfie',  
 'Take Selfie for starting Questionnaire':  
 'Take Selfie for starting Questionnaire',  
 'Take Selfie': 'Take Selfie',  
 'Take / Upload': 'Take / Upload',  
 'Upload from Gallery': 'Upload from Gallery',  
 'Next': 'Next',  
 'Scan Your ID': 'Scan Your ID',  
 'Personal Details\nConfirmation': 'Personal Details\nConfirmation',  
 'Mobile & Email\nConfirmation': 'Mobile & Email\nConfirmation',  
 'Enter Verification\nCode': 'Enter Verification\nCode',  
 'Scan Your License': 'Scan Your License',  
 'License Details\nConfirmation': 'License Details\nConfirmation',  
 'Confirm': 'Confirm',  
 'Verify': 'Verify',  
 'Back': 'Back',  
 'Please confirm your\nPersonal Information':  
 'Please confirm your\nPersonal Information',  
 'ID Number': 'ID Number',  
 'ID Number not be empty!': 'ID Number not be empty!',  
 'Full Name': 'Full Name',  
 'Full Name not be empty!': 'Full Name not be empty!',  
 'Nationality': 'Nationality',  
 'Nationality not be empty!': 'Nationality not be empty!',  
 'Scan the back of your ': 'Scan the back of your ',  
 'Emirates ID': 'Emirates ID',  
 'Start Scanning': 'Start Scanning',  
 'Card Number': 'Card Number',  
 'Card Number not be empty!': 'Card Number not be empty!',  
 'Please Place the back of your\nTrade License inside the frame':  
 'Please Place the back of your\nTrade License inside the frame',  
 'Scan License': 'Scan License',  
 'Please Place the back of your\nEmirates ID inside the frame':  
 'Please Place the back of your\nEmirates ID inside the frame',  
 'Scan Card': 'Scan Card',  
 'Date of Birth': 'Date of Birth',  
 'Date of Birth not be empty!': 'Date of Birth not be empty!',  
 'Expiry Date': 'Expiry Date',  
 'Expiry Date not be empty!': 'Expiry Date not be empty!',  
 'Gender': 'Gender',  
 'Male': 'Male',  
 'Female': 'Female',  
 'Other': 'Other',  
 'Gender not be empty!': 'Gender not be empty!',  
 'Please provide your Mobile Phone Number and Email to continue':  
 'Please provide your Mobile Phone Number and Email to continue',  
 'Phone Number': 'Phone Number',  
 'Email': 'Email',  
 'Please enter the OTP Code\nwe sent you':  
 'Please enter the OTP Code\nwe sent you',  
 'I did not receive the OTP': 'I did not receive the OTP',  
 'Resend Code': 'Resend Code',  
 'Please confirm your\nLicense Information':  
 'Please confirm your\nLicense Information',  
 'License No': 'License No',  
 'License No not be empty!': 'License No not be empty!',  
 'Trade Name': 'Trade Name',  
 'Trade Name not be empty!': 'Trade Name not be empty!',  
 'Scan the back of your\n': 'Scan the back of your\n',  
 'Trade License': 'Trade License',  
 'WellDone': 'WellDone',  
 'Congratulations': 'Congratulations',  
 'You have successfully registered\nyour ALGORITHMI account':  
 'You have successfully registered\nyour ALGORITHMI account',  
 'You have successfully added\nyour':  
 'You have successfully added\nyour',  
 'You\'re upload file and Submitted\nfor review':  
 'You\'re upload file and Submitted\nfor review',  
 'Your survey has been Finished and\nSubmitted for review':  
 'Your survey has been Finished and\nSubmitted for review',  
 'Go to Dashboard': 'Go to Dashboard',  
 'Dashboard': 'Dashboard',  
 'Wallet': 'Wallet',  
 'Settings': 'Settings',  
 'Muhammad Usman': 'Muhammad Usman',  
 'Verified Account': 'Verified Account',  
 'Current location: ': 'Current location: ',  
 'Al Barai, Dubai': 'Al Barai, Dubai',  
 'Account Type: ': 'Account Type: ',  
 'Shop Keeper': 'Shop Keeper',  
 'Member Status: ': 'Member Status: ',  
 'Bronze': 'Bronze',  
 'Questioners: ': 'Questioners: ',  
 '3/5 Approved': '3/5 Approved',  
 'Questionnaire': 'Questionnaire',  
 '1) Does the location have a POS system.\n2) # of counters.\n3) Sales of tobacco.\n4) Attach image of tobacco shelf.\n5) Ask shop keep top selling brands.\n6) Sales of energy drinks.':  
 '1) Does the location have a POS system.\n2) # of counters.\n3) Sales of tobacco.\n4) Attach image of tobacco shelf.\n5) Ask shop keep top selling brands.\n6) Sales of energy drinks.',  
 'Accept': 'Accept',  
 'Cancel': 'Cancel',  
 'Start New Survey': 'Start New Survey',  
 'Last Survey Conducted on 26 Nov 2022':  
 'Last Survey Conducted on 26 Nov 2022',  
 'Upload Invoices': 'Upload Invoices',  
 'Last Invoice Uploaded on 26 Nov 2022':  
 'Last Invoice Uploaded on 26 Nov 2022',  
 'Upload Stock Report': 'Upload Stock Report',  
 'Last Report Uploaded on 26 Nov 2022':  
 'Last Report Uploaded on 26 Nov 2022',  
 'Let’s Start...': 'Let’s Start...',  
 'Quick Supermarket, Majan': 'Quick Supermarket, Majan',  
 'Your slot has been blocked for\n60 min':  
 'Your slot has been blocked for\n60 min',  
 'Start Survey Now!': 'Start Survey Now!',  
 'Upload File': 'Upload File',  
 'Select File': 'Select File',  
 'Question': 'Question',  
 'Does the location have a POS system?':  
 'Does the location have a POS system?',  
 'Total Numbers of cash counters?': 'Total Numbers of cash counters?',  
 'Do they sale tobacco?': 'Do they sale tobacco?',  
 'Attach image of tobacco shelf.': 'Attach image of tobacco shelf.',  
 'Ask shopkeeper top selling brands?':  
 'Ask shopkeeper top selling brands?',  
 'Take Picture': 'Take Picture',  
 'Yes': 'Yes',  
 'No': 'No',  
 'Select Brand': 'Select Brand',  
 'Nike': 'Nike',  
 'Hermes': 'Hermes',  
 'Louis Vuitton': 'Louis Vuitton',  
 'Select Option': 'Select Option',  
 'Select Amount': 'Select Amount',  
 'Upload Image': 'Upload Image',  
 'Sign Out': 'Sign Out',  
 'Language': 'Language',  
 'Preferences': 'Preferences',  
 'Rate the App': 'Rate the App',  
 'Help': 'Help',  
 'Support': 'Support',  
 'Rewards': 'Rewards',  
 'You have 568 points': 'You have 568 points',  
 'Benefits': 'Benefits',  
 'Notifications': 'Notifications',  
 'Accounts': 'Accounts',  
 'Personal Information': 'Personal Information',  
 'Your Account': 'Your Account',  
 'Profile': 'Profile',  
 'Date Of Birth': 'Date Of Birth',  
 'United Arab Emirates': 'United Arab Emirates',  
 'Saved Payment Methods': 'Saved Payment Methods',  
 'Add New': 'Add New',  
 'Add': 'Add',  
 'Add Payment Method': 'Add Payment Method',  
 'First Abu Dhabi Bank': 'First Abu Dhabi Bank',  
 'Dubai Islamic Bank': 'Dubai Islamic Bank',  
 'Etisalat Wallet': 'Etisalat Wallet',  
 'Magnati': 'Magnati',  
 'Bank Account': 'Bank Account',  
 'E-Wallet': 'E-Wallet',  
 'Today\'s Notification': 'Today\'s Notification',  
 'Approved Review': 'Approved Review',  
 'Your Given Review about Shop is approved from Admin.':  
 'Your Given Review about Shop is approved from Admin.',  
 'Send via crypto network': 'Send via crypto network',  
 'Send via email/ Phone/ Pay ID': 'Send via email/ Phone/ Pay ID',  
 'send to a known crypto address\nvia crypto network':  
 'send to a known crypto address\nvia crypto network',  
 'Recipient must be a Algorthmi user':  
 'Recipient must be a Algorthmi user',  
 'Coins': 'Coins',  
 'Cash': 'Cash',  
 'You currently have': 'You currently have',  
 'Link Account': 'Link Account',  
 'Transfer to Bank/E-Wallet': 'Transfer to Bank/E-Wallet',  
 'Withdraw': 'Withdraw',  
 'Transfer': 'Transfer',  
 'Withdrawal Money': 'Withdrawal Money',  
 'Al Ansari Exchange': 'Al Ansari Exchange',  
  
  
 },  
 'ar\_SA': {  
 'Select your preferred language': 'اختر لغتك المفضلة',  
 'Login': 'تسجيل الدخول',  
 'Enter Your Email': 'أدخل بريدك الإلكتروني',  
 'Please enter a valid email': 'يرجى إدخال البريد الإلكتروني الصحيح',  
 'Enter Your Password': 'ادخل رقمك السري',  
 'Password not be empty!': 'كلمة المرور ليست فارغة!',  
 'Password must be greater than 5 character!':  
 'يجب أن تكون كلمة المرور أكبر من 5 أحرف!',  
 'Forgot Password?': 'هل نسيت كلمة السر؟',  
 'Forgot Password': 'هل نسيت كلمة السر',  
 'Don’t have an account ? ': 'ليس لديك حساب؟ ',  
 'Sign Up': 'اشتراك',  
 'Don’t worry! It happens. Please enter the email address associated with your account.':  
 'لا تقلق! يحدث ذلك. الرجاء إدخال عنوان البريد الإلكتروني المرتبط بحسابك.',  
 'Submit': 'يُقدِّم',  
 'Reset Password': 'إعادة تعيين كلمة المرور',  
 'Your new password must be different from previous used passwords.':  
 'يجب أن تكون كلمة مرورك الجديدة مختلفة عن كلمات المرور السابقة المستخدمة.',  
 'Password': 'كلمة المرور',  
 'Confirm Password': 'تأكيد كلمة المرور',  
 'Select Role': 'حدد الدور',  
 'Freelancers': 'لحسابهم الخاص',  
 'Shop Owner': 'صاحب محل',  
 'Already have an account ? ': 'هل لديك حساب؟ ',  
 'Upload Selfie': 'تحميل صورة شخصية',  
 'Take Selfie for starting Questionnaire':  
 'التقط صورة شخصية مع ملصق اسم المتجر لبدء الاستبيان',  
 'Take Selfie': 'خذ صورة شخصية',  
 'Take / Upload': 'خذ / تحميل',  
 'Upload from Gallery': 'تحميل من المعرض',  
 'Next': 'التالي',  
 'Scan Your ID': 'امسح هويتك ضوئيًا',  
 'Personal Details\nConfirmation': 'التفاصيل الشخصية \n تأكيد',  
 'Mobile & Email\nConfirmation': 'الجوال والبريد\n الإلكتروني التأكيد',  
 'Enter Verification\nCode': 'أدخل التحقق \n الرمز',  
 'Scan Your License': 'مسح الترخيص\n الخاص بك',  
 'License Details\nConfirmation': 'تفاصيل الترخيص \n تأكيد',  
 'Confirm': 'يتأكد',  
 'Verify': 'يؤكد',  
 'Back': 'خلف',  
 'Please confirm your\nPersonal Information':  
 'الرجاء تأكيد المعلومات \nالشخصية الخاصة بك',  
 'ID Number': 'رقم الهوية',  
 'ID Number not be empty!': 'رقم الهوية ليس فارغا!',  
 'Full Name': 'الاسم الكامل',  
 'Full Name not be empty!': 'الاسم الكامل ليس فارغًا!',  
 'Nationality': 'جنسية',  
 'Nationality not be empty!': 'الجنسية لا تكون فارغة!',  
 'Scan the back of your ': 'امسح الجزء الخلفي من ملف',  
 'Emirates ID': 'هويه الإمارات',  
 'Start Scanning': 'ابدأ المسح',  
 'Card Number': 'رقم البطاقة',  
 'Card Number not be empty!': 'رقم البطاقة غير فارغ!',  
 'Please Place the back of your\nTrade License inside the frame':  
 'الرجاء وضع الجزء الخلفي من\n الرخصة التجارية الخاصة بك داخل الإطار',  
 'Scan License': 'رخصة الفحص',  
 'Please Place the back of your\nEmirates ID inside the frame':  
 'يرجى وضع ظهر بطاقة الهوية \nالإماراتية الخاصة بك داخل الإطار',  
 'Scan Card': 'مسح البطاقة',  
 'Date of Birth': 'تاريخ الميلاد',  
 'Date of Birth not be empty!': 'تاريخ الميلاد يجب ألا يكون فارغًا!',  
 'Expiry Date': 'تاريخ الانتهاء',  
 'Expiry Date not be empty!':  
 'لا يمكن ترك تاريخ انتهاء الصلاحية فارغًا!',  
 'Gender': 'جنس',  
 'Male': 'ذكر',  
 'Female': 'أنثى',  
 'Other': 'آخر',  
 'Gender not be empty!': 'لا يكون الجنس فارغًا!',  
 'Please provide your Mobile Phone Number and Email to continue':  
 'يرجى تقديم رقم هاتفك المحمول والبريد الإلكتروني للمتابعة',  
 'Phone Number': 'رقم التليفون',  
 'Email': 'بريد إلكتروني',  
 'Please enter the OTP Code\nwe sent you':  
 'الرجاء إدخال رمز OTP \n الذي أرسلناه إليك',  
 'I did not receive the OTP': 'لم أحصل على OTP',  
 'Resend Code': 'أعد إرسال الرمز',  
 'Please confirm your\nLicense Information':  
 'الرجاء تأكيد \n معلومات الترخيص الخاصة بك',  
 'License No': 'رقم الرخصة',  
 'License No not be empty!': 'الرخصة لا لا تكون فارغة!',  
 'Trade Name': 'اسم تجاري',  
 'Trade Name not be empty!': 'الاسم التجاري يجب ألا يكون فارغًا!',  
 'Scan the back of your\n': 'امسح الجزء الخلفي من ملف\n',  
 'Trade License': 'الرخصة التجارية',  
 'WellDone': 'أحسنت',  
 'Congratulations': 'تهانينا',  
 'You have successfully registered\nyour ALGORITHMI account':  
 'لقد سجلت بنجاح \n حساب ALGORITHMI الخاص بك',  
 'You have successfully added\nyour': 'لقد نجحت في إضافة \n الخاص بك',  
 'You\'re upload file and Submitted\nfor review':  
 'تقوم بتحميل الملف وإرساله \n للمراجعة',  
 'Your survey has been Finished and\nSubmitted for review':  
 'تم الانتهاء من الاستبيان الخاص بك \n وتم إرساله للمراجعة',  
 'Go to Dashboard': 'الذهاب إلى لوحة القيادة',  
 'Dashboard': 'لوحة القيادة',  
 'Wallet': 'محفظة',  
 'Settings': 'جلسة',  
 'Muhammad Usman': 'محمد عثمان',  
 'Verified Account': 'حساب تم التحقق منه',  
 'Current location: ': 'الموقع الحالي: ',  
 'Al Barai, Dubai': 'البرعي ، دبي',  
 'Account Type: ': 'نوع الحساب: ',  
 'Shop Keeper': 'حارس متجر',  
 'Member Status: ': 'حالة العضو: ',  
 'Bronze': 'برونزية',  
 'Questioners: ': 'المستجوبون: ',  
 '3/5 Approved': '3/5 معتمد',  
 'Questionnaire': 'استبيان',  
 '1) Does the location have a POS system.\n2) # of counters.\n3) Sales of tobacco.\n4) Attach image of tobacco shelf.\n5) Ask shop keep top selling brands.\n6) Sales of energy drinks.':  
 '1) هل الموقع به نظام نقاط البيع. \n 2) عدد العدادات. \n 3) بيع التبغ. \n 4) إرفاق صورة لرف التبغ. \n 5) اسأل المتجر عن أفضل العلامات التجارية مبيعًا. \n 6) بيع مشروبات الطاقة.',  
 'Accept': 'يقبل',  
 'Cancel': 'يلغي',  
 'Start New Survey': 'بدء استبيان جديد',  
 'Last Survey Conducted on 26 Nov 2022':  
 'تم إجراء آخر مسح في 26 نوفمبر 2022',  
 'Upload Invoices': 'تحميل الفواتير',  
 'Last Invoice Uploaded on 26 Nov 2022':  
 'تم تحميل آخر فاتورة في 26 نوفمبر 2022',  
 'Upload Stock Report': 'تحميل تقرير المخزون',  
 'Last Report Uploaded on 26 Nov 2022':  
 'تم تحميل التقرير الأخير في 26 نوفمبر 2022',  
 'Let’s Start...': 'لنبدأ...',  
 'Quick Supermarket, Majan': 'سوبر ماركت كويك ، مجان',  
 'Your slot has been blocked for\n60 min':  
 'تم حظر الفتحة الخاصة بك لمدة \n 60 دقيقة',  
 'Start Survey Now!': 'ابدأ الاستبيان الآن!',  
 'Upload File': 'رفع ملف',  
 'Select File': 'حدد ملف',  
 'Question': 'سؤال',  
 'Does the location have a POS system?':  
 'هل الموقع به نظام نقاط البيع؟',  
 'Total Numbers of cash counters?': 'العدد الإجمالي للعدادات النقدية؟',  
 'Do they sale tobacco?': 'هل يبيعون التبغ؟',  
 'Attach image of tobacco shelf.': 'إرفاق صورة رف التبغ.',  
 'Ask shopkeeper top selling brands?':  
 'اسأل صاحب متجر الماركات الأكثر مبيعًا؟',  
 'Take Picture': 'التقط صورة',  
 'Yes': 'نعم',  
 'No': 'لا',  
 'Select Brand': 'اختر الماركة',  
 'Nike': 'نايك',  
 'Hermes': 'هيرميس',  
 'Louis Vuitton': 'لويس فيتون',  
 'Select Option': 'حدد خيار',  
 'Select Amount': 'حدد المبلغ',  
 'Upload Image': 'تحميل الصور',  
 'Sign Out': 'خروج',  
 'Language': 'لغة',  
 'Preferences': 'التفضيلات',  
 'Rate the App': 'قيم التطبيق',  
 'Help': 'يساعد',  
 'Support': 'يدعم',  
 'Rewards': 'المكافآت',  
 'You have 568 points': 'لديك 568 نقطة',  
 'Benefits': 'فوائد',  
 'Notifications': 'إشعارات',  
 'Accounts': 'حسابات',  
 'Personal Information': 'معلومات شخصية',  
 'Your Account': 'الحساب الخاص بك',  
 'Profile': 'حساب تعريفي',  
 'Date Of Birth': 'تاريخ الميلاد',  
 'United Arab Emirates': 'الإمارات العربية المتحدة',  
 'Saved Payment Methods': 'طرق الدفع المحفوظة',  
 'Add New': 'اضف جديد',  
 'Add': 'يضيف',  
 'Add Payment Method': 'إضافة طريقة دفع',  
 'First Abu Dhabi Bank': 'بنك أبوظبي الأول',  
 'Dubai Islamic Bank': 'بنك دبي الإسلامي',  
 'Etisalat Wallet': 'اتصالات واللي',  
 'Magnati': 'ماجناتي',  
 'Bank Account': 'حساب البنك',  
 'E-Wallet': 'المحفظة الإلكترونية',  
 'Today\'s Notification': 'إشعار اليوم',  
 'Approved Review': 'مراجعة معتمدة',  
 'Your Given Review about Shop is approved from Admin.':  
 'تمت الموافقة على مراجعتك المقدمة حول Shop من المسؤول.',  
 'Send via crypto network': 'إرسال عبر شبكة التشفير',  
 'Send via email/ Phone/ Pay ID': 'إرسال عبر البريد الإلكتروني / الهاتف / معرف الدفع',  
 'send to a known crypto address\nvia crypto network':  
 'إرسال إلى عنوان تشفير معروف \n عبر شبكة تشفير',  
 'Recipient must be a Algorthmi user':  
 'يجب أن يكون المستلم من مستخدمي الجورتمي',  
 'Coins': 'عملات معدنية',  
 'Cash': 'نقدي',  
 'You currently have': 'أنت حاليا تملك',  
 'Link Account': 'حساب الارتباط',  
 'Transfer to Bank/E-Wallet': 'التحويل إلى البنك / المحفظة الإلكترونية',  
 'Withdraw': 'ينسحب',  
 'Transfer': 'تحويل',  
 'Withdrawal Money': 'سحب الأموال',  
 'Al Ansari Exchange': 'الانصاري للصرافة',  
 },  
 'fr\_FR': {  
 'Select your preferred language':  
 'Sélectionnez votre langue préférée',  
 'Login': 'Connexion',  
 'Enter Your Email': 'Entrer votre Email',  
 'Please enter a valid email': 'veuillez entrer un email valide',  
 'Enter Your Password': 'Tapez votre mot de passe',  
 'Password not be empty!': 'Le mot de passe ne doit pas être vide !',  
 'Password must be greater than 5 character!':  
 'Le mot de passe doit comporter plus de 5 caractères!',  
 'Forgot Password?': 'Mot de passe oublié?',  
 'Forgot Password': 'Mot de passe oublié',  
 'Don’t have an account ? ': 'Vous n\'avez pas de compte ? ',  
 'Sign Up': 'S\'inscrire',  
 'Don’t worry! It happens. Please enter the email address associated with your account.':  
 'Ne t\'inquiète pas! Ça arrive. Veuillez saisir l\'adresse e-mail associée à votre compte.',  
 'Submit': 'Soumettre',  
 'Reset Password': 'Réinitialiser le mot de passe',  
 'Your new password must be different from previous used passwords.':  
 'Votre nouveau mot de passe doit être différent des mots de passe utilisés précédemment.',  
 'Password': 'mot de passe',  
 'Confirm Password': 'Confirmez le mot de passe',  
 'Select Role': 'Sélectionnez un rôle',  
 'Freelancers': 'Pigistes',  
 'Shop Owner': 'Propriétaire de la boutique',  
 'Already have an account ? ': 'Vous avez déjà un compte? ',  
 'Upload Selfie': 'Télécharger un selfie',  
 'Take Selfie for starting Questionnaire':  
 'Prenez Selfie avec l\'étiquette du nom de la boutique pour commencer le questionnaire',  
 'Take Selfie': 'Prendre un selfie',  
 'Take / Upload': 'Prendre / Télécharger',  
 'Upload from Gallery': 'Télécharger depuis la galerie',  
 'Next': 'Suivant',  
 'Scan Your ID': 'Scannez votre pièce d\'identité',  
 'Personal Details\nConfirmation': 'Détails personnels\nConfirmation',  
 'Mobile & Email\nConfirmation': 'Mobile et e-mail\nConfirmation',  
 'Enter Verification\nCode': 'Entrez le code\nde vérification',  
 'Scan Your License': 'Scannez votre \nlicence',  
 'License Details\nConfirmation':  
 'Détails de la licence\nConfirmation',  
 'Confirm': 'Confirmer',  
 'Verify': 'Vérifier',  
 'Back': 'Dos',  
 'Please confirm your\nPersonal Information':  
 'Veuillez confirmer vos \n informations personnelles',  
 'ID Number': 'Numéro d\'identification',  
 'ID Number not be empty!':  
 'Le numéro d\'identification ne doit pas être vide!',  
 'Full Name': 'Nom et prénom',  
 'Full Name not be empty!': 'Le nom complet n\'est pas vide!',  
 'Nationality': 'Nationalité',  
 'Nationality not be empty!': 'La nationalité ne doit pas être vide!',  
 'Scan the back of your ': 'Scannez le dos de votre ',  
 'Emirates ID': 'carte d\'identité Emirates',  
 'Start Scanning': 'Lancer la numérisation',  
 'Card Number': 'Numéro de carte',  
 'Card Number not be empty!':  
 'Le numéro de carte ne doit pas être vide!',  
 'Please Place the back of your\nTrade License inside the frame':  
 'Veuillez placer le dos de \nvotre licence commerciale à l\'intérieur du cadre',  
 'Scan License': 'Licence de numérisation',  
 'Please Place the back of your\nEmirates ID inside the frame':  
 'Veuillez placer le dos de votre carte \nd\'identité Emirates à l\'intérieur du cadre',  
 'Scan Card': 'Numériser la carte',  
 'Date of Birth': 'Date de naissance',  
 'Date of Birth not be empty!':  
 'La date de naissance n\'est pas vide!',  
 'Expiry Date': 'Date d\'expiration',  
 'Expiry Date not be empty!': 'La date d\'expiration n\'est pas vide!',  
 'Gender': 'Genre',  
 'Male': 'Homme',  
 'Female': 'Femme',  
 'Other': 'Autre',  
 'Gender not be empty!': 'Le genre n\'est pas vide !',  
 'Please provide your Mobile Phone Number and Email to continue':  
 'Veuillez fournir votre numéro de téléphone portable et votre e-mail pour continuer',  
 'Phone Number': 'Numéro de téléphone',  
 'Email': 'E-mail',  
 'Please enter the OTP Code\nwe sent you':  
 'Veuillez entrer le code OTP\nque nous vous avons envoyé',  
 'I did not receive the OTP': 'Je n\'ai pas reçu l\'OTP',  
 'Resend Code': 'Renvoyer le code',  
 'Please confirm your\nLicense Information':  
 'Veuillez confirmer vos\ninformations de licence',  
 'License No': 'License No',  
 'License No not be empty!': 'N ° de licence',  
 'Trade Name': 'Trade Name',  
 'Trade Name not be empty!': 'Licence Non pas être vide !',  
 'Scan the back of your\n': 'Nom commercial\n',  
 'Trade License': 'Le nom commercial ne doit pas être vide!',  
 'WellDone': 'Bien joué',  
 'Congratulations': 'Toutes nos félicitations',  
 'You have successfully registered\nyour ALGORITHMI account':  
 'Vous avez enregistré avec succès \n votre compte ALGORITHM',  
 'You have successfully added\nyour':  
 'Vous avez ajouté avec succès \n votre',  
 'You\'re upload file and Submitted\nfor review':  
 'Vous téléchargez le fichier et l\'envoyez \n pour examen',  
 'Your survey has been Finished and\nSubmitted for review':  
 'Votre enquête a été terminée et \n soumise pour examen',  
 'Go to Dashboard': 'Aller au tableau de bord',  
 'Dashboard': 'Tableau de bord',  
 'Wallet': 'Portefeuille',  
 'Settings': 'Paramètre',  
 'Muhammad Usman': 'Mohammed Ousman',  
 'Verified Account': 'Compte vérifié',  
 'Current location: ': 'Localisation actuelle: ',  
 'Al Barai, Dubai': 'Al Baraï, Dubaï',  
 'Account Type: ': 'Type de compte: ',  
 'Shop Keeper': 'Magasinier',  
 'Member Status: ': 'Statut de membre : ',  
 'Bronze': 'Bronze',  
 'Questioners: ': 'Interrogateurs : ',  
 '3/5 Approved': '3/5 Approuvé',  
 'Questionnaire': 'Questionnaire',  
 '1) Does the location have a POS system.\n2) # of counters.\n3) Sales of tobacco.\n4) Attach image of tobacco shelf.\n5) Ask shop keep top selling brands.\n6) Sales of energy drinks.':  
 '1) L\'emplacement dispose-t-il d\'un système de point de vente. \n 2) # de compteurs. \n 3) Ventes de tabac. \n 4) Joignez l\'image de l\'étagère à tabac. \n 5) Demandez au magasin de conserver les marques les plus vendues. \n 6) Ventes de boissons énergisantes.',  
 'Accept': 'Accepter',  
 'Cancel': 'Annuler',  
 'Start New Survey': 'Commencer une nouvelle enquête',  
 'Last Survey Conducted on 26 Nov 2022':  
 'Dernière enquête réalisée le 26 novembre 2022',  
 'Upload Invoices': 'Télécharger des factures',  
 'Last Invoice Uploaded on 26 Nov 2022':  
 'Dernière facture téléchargée le 26 novembre 2022',  
 'Upload Stock Report': 'Télécharger le rapport de stock',  
 'Last Report Uploaded on 26 Nov 2022':  
 'Dernier rapport mis en ligne le 26 nov. 2022',  
 'Let’s Start...': 'Commençons...',  
 'Quick Supermarket, Majan': 'Supermarché rapide, Majan',  
 'Your slot has been blocked for\n60 min':  
 'Votre emplacement a été bloqué pendant \n60 min',  
 'Start Survey Now!': 'Commencez l\'enquête maintenant!',  
 'Upload File': 'Téléverser un fichier',  
 'Select File': 'Choisir le dossier',  
 'Question': 'Question',  
 'Does the location have a POS system?':  
 'L\'emplacement dispose-t-il d\'un système de point de vente?',  
 'Total Numbers of cash counters?': 'Nombre total de guichets?',  
 'Do they sale tobacco?': 'Vendent-ils du tabac ?',  
 'Attach image of tobacco shelf.':  
 'Joindre l\'image de l\'étagère à tabac.',  
 'Ask shopkeeper top selling brands?':  
 'Demandez aux commerçants les marques les plus vendues?',  
 'Take Picture': 'Prendre une photo',  
 'Yes': 'Oui',  
 'No': 'Non',  
 'Select Brand': 'Sélectionner la marque',  
 'Nike': 'Nike',  
 'Hermes': 'Hermès',  
 'Louis Vuitton': 'Louis Vuitton',  
 'Select Option': 'Sélectionner une option',  
 'Select Amount': 'Sélectionnez le montant',  
 'Upload Image': 'Télécharger une image',  
 'Sign Out': 'Se déconnecter',  
 'Language': 'Langue',  
 'Preferences': 'Préférences',  
 'Rate the App': 'Évaluer l\'application',  
 'Help': 'Aider',  
 'Support': 'Soutien',  
 'Rewards': 'Récompenses',  
 'You have 568 points': 'Vous avez 568 points',  
 'Benefits': 'Avantages',  
 'Notifications': 'Avis',  
 'Accounts': 'Comptes',  
 'Personal Information': 'Informations personnelles',  
 'Your Account': 'Votre compte',  
 'Profile': 'Profil',  
 'Date Of Birth': 'Date de naissance',  
 'United Arab Emirates': 'Emirats Arabes Unis',  
 'Saved Payment Methods': 'Méthodes de paiement enregistrées',  
 'Add New': 'Ajouter nouveau',  
 'Add': 'Ajouter',  
 'Add Payment Method': 'Ajouter un mode de paiement',  
 'First Abu Dhabi Bank': 'Première banque d\'Abu Dhabi',  
 'Dubai Islamic Bank': 'Banque islamique de Dubaï',  
 'Etisalat Wallet': 'Étisalat Walle',  
 'Magnati': 'Magnati',  
 'Bank Account': 'Compte bancaire',  
 'E-Wallet': 'Portefeuille électronique',  
 'Today\'s Notification': 'Notification d\'aujourd\'hui',  
 'Approved Review': 'Examen approuvé',  
 'Your Given Review about Shop is approved from Admin.':  
 'Votre avis sur la boutique est approuvé par l\'administrateur.',  
 'Send via crypto network': 'Envoyer via le réseau crypto',  
 'Send via email/ Phone/ Pay ID': 'Envoyer par e-mail/Téléphone/Pay ID',  
 'send to a known crypto address\nvia crypto network':  
 'envoyer à une adresse crypto connue \n via un réseau crypto',  
 'Recipient must be a Algorthmi user':  
 'Le destinataire doit être un utilisateur d\'Algorthmi',  
 'Coins': 'Pièces de monnaie',  
 'Cash': 'Espèces',  
 'You currently have': 'Tu as présentement',  
 'Link Account': 'Lien de compte',  
 'Transfer to Bank/E-Wallet': 'Virement bancaire / portefeuille électronique',  
 'Withdraw': 'Retirer',  
 'Transfer': 'Transfert',  
 'Withdrawal Money': 'Retrait d\'argent',  
 'Al Ansari Exchange': 'Échange Al Ansari',  
 },  
 'hi\_IN': {  
 'Select your preferred language': 'अपनी पसंदीदा भाषा का चयन करें',  
 'Login': 'लॉग इन करें',  
 'Enter Your Email': 'अपना ईमेल दर्ज करें',  
 'Please enter a valid email': 'कृपया एक मान्य ईमेल दर्ज करें',  
 'Enter Your Password': 'अपना कूटशब्द भरें',  
 'Password not be empty!': 'पासवर्ड खाली न हो!',  
 'Password must be greater than 5 character!':  
 'पासवर्ड 5 कैरेक्टर से बड़ा होना चाहिए!',  
 'Forgot Password?': 'पासवर्ड भूल गए?',  
 'Forgot Password': 'पासवर्ड भूल गए',  
 'Don’t have an account ? ': 'खाता नहीं है? ',  
 'Sign Up': 'साइन अप करें',  
 'Don’t worry! It happens. Please enter the email address associated with your account.':  
 'चिंता मत करो! ऐसा होता है। कृपया अपने खाते से संबद्ध ईमेल पता दर्ज करें।',  
 'Submit': 'जमा करना',  
 'Reset Password': 'पासवर्ड रीसेट',  
 'Your new password must be different from previous used passwords.':  
 'आपका नया पासवर्ड पिछले इस्तेमाल किए गए पासवर्ड से अलग होना चाहिए।',  
 'Password': 'पासवर्ड',  
 'Confirm Password': 'पासवर्ड की पुष्टि कीजिये',  
 'Select Role': 'भूमिका का चयन करें',  
 'Freelancers': 'स्वतंत्र',  
 'Shop Owner': 'दुकान का मालिक',  
 'Already have an account ? ':  
 'क्या आपके पास पहले से एक खाता मौजूद है? ',  
 'Upload Selfie': 'सेल्फी अपलोड करें',  
 'Take Selfie for starting Questionnaire':  
 'प्रश्नावली शुरू करने के लिए दुकान के नाम के लेबल के साथ सेल्फी लें',  
 'Take Selfie': 'सेल्फी लो',  
 'Take / Upload': 'लो / अपलोड करो',  
 'Upload from Gallery': 'गैलरी से अपलोड करें',  
 'Next': 'अगला',  
 'Scan Your ID': 'अपनी आईडी स्कैन करें',  
 'Personal Details\nConfirmation': 'व्यक्तिगत विवरण\nपुष्टि',  
 'Mobile & Email\nConfirmation': 'मोबाइल और ईमेल\nपुष्टि',  
 'Enter Verification\nCode': 'सत्यापन\nकोड दर्ज करें',  
 'Scan Your License': 'अपना लाइसेंस \nस्कैन करें',  
 'License Details\nConfirmation': 'लाइसेंस विवरण\nपुष्टि',  
 'Confirm': 'पुष्टि करना',  
 'Verify': 'सत्यापित करना',  
 'Back': 'पीछे',  
 'Please confirm your\nPersonal Information':  
 'कृपया अपनी \n व्यक्तिगत जानकारी की पुष्टि करेंं',  
 'ID Number': 'आईडी नंबर',  
 'ID Number not be empty!': 'आईडी नंबर खाली न हो!',  
 'Full Name': 'पूरा नाम',  
 'Full Name not be empty!': 'पूरा नाम खाली नहीं है!',  
 'Nationality': 'राष्ट्रीयता',  
 'Nationality not be empty!': 'राष्ट्रीयता खाली नहीं!',  
 'Scan the back of your ': 'अपने पीछे स्कैन करें ',  
 'Emirates ID': 'अमीरात आईडी',  
 'Start Scanning': 'स्कैन करना प्रारंभ करें',  
 'Card Number': 'कार्ड संख्या',  
 'Card Number not be empty!': 'कार्ड नंबर खाली न हो!',  
 'Please Place the back of your\nTrade License inside the frame':  
 'कृपया अपने ट्रेड लाइसेंस के \nपिछले हिस्से को फ्रेम के अंदर रखें',  
 'Scan License': 'स्कैन लाइसेंस',  
 'Please Place the back of your\nEmirates ID inside the frame':  
 'कृपया अपनी अमीरात आईडी के \nपिछले हिस्से को फ्रेम के अंदर रखें',  
 'Scan Card': 'स्कैन कार्ड',  
 'Date of Birth': 'जन्म की तारीख',  
 'Date of Birth not be empty!': 'जन्मतिथि खाली नहीं है!',  
 'Expiry Date': 'समाप्ति तिथि',  
 'Expiry Date not be empty!': 'समाप्ति तिथि खाली नहीं है!',  
 'Gender': 'लिंग',  
 'Male': 'नर',  
 'Female': 'महिला',  
 'Other': 'अन्य',  
 'Gender not be empty!': 'लिंग खाली नहीं है!',  
 'Please provide your Mobile Phone Number and Email to continue':  
 'जारी रखने के लिए कृपया अपना मोबाइल फ़ोन नंबर और ईमेल प्रदान करें',  
 'Phone Number': 'फ़ोन नंबर',  
 'Email': 'ईमेल',  
 'Please enter the OTP Code\nwe sent you':  
 'कृपया ओटीपी कोड दर्ज करें\nहमने आपको भेजा है',  
 'I did not receive the OTP': 'मुझे ओटीपी प्राप्त नहीं हुआ',  
 'Resend Code': 'पुन: कोड भेजे',  
 'Please confirm your\nLicense Information':  
 'कृपया अपनी \nलाइसेंस जानकारी की पुष्टि करें',  
 'License No': 'अनुज्ञापत्र क्रमांक',  
 'License No not be empty!': 'लाइसेंस नहीं खाली नहीं!',  
 'Trade Name': 'Trade Name',  
 'Trade Name not be empty!': 'व्यापरिक नाम',  
 'Scan the back of your\n': 'ट्रेड नाम खाली न हो!\n',  
 'Trade License': 'अपने पीछे स्कैन करें',  
 'WellDone': 'बहुत अच्छा',  
 'Congratulations': 'बधाई हो',  
 'You have successfully registered\nyour ALGORITHMI account':  
 'आपने \n अपना ALGORITHMI खाता सफलतापूर्वक पंजीकृत कर लिया है',  
 'You have successfully added\nyour':  
 'आपने \n सफलतापूर्वक जोड़ लिया है',  
 'You\'re upload file and Submitted\nfor review':  
 'आप फ़ाइल अपलोड कर रहे हैं और \n समीक्षा के लिए सबमिट कर दिया गया है',  
 'Your survey has been Finished and\nSubmitted for review':  
 'आपका सर्वे पूरा हो चुका है और \n समीक्षा के लिए सबमिट कर दिया गया है',  
 'Go to Dashboard': 'डैशबोर्ड पर जाएँ',  
 'Dashboard': 'डैशबोर्ड',  
 'Wallet': 'बटुआ',  
 'Settings': 'सेटिंग',  
 'Muhammad Usman': 'मुहम्मद उस्मान',  
 'Verified Account': 'सत्यापित खाते',  
 'Current location: ': 'वर्तमान स्थान: ',  
 'Al Barai, Dubai': 'अल बरई, दुबई',  
 'Account Type: ': 'खाते का प्रकार: ',  
 'Shop Keeper': 'दुकानदारr',  
 'Member Status: ': 'सदस्य स्थिति: ',  
 'Bronze': 'पीतल',  
 'Questioners: ': 'प्रश्नकर्ता: ',  
 '3/5 Approved': '3/5 स्वीकृत',  
 'Questionnaire': 'प्रश्नावली',  
 '1) Does the location have a POS system.\n2) # of counters.\n3) Sales of tobacco.\n4) Attach image of tobacco shelf.\n5) Ask shop keep top selling brands.\n6) Sales of energy drinks.':  
 '1) क्या स्थान में पीओएस सिस्टम है। \n 2) # काउंटरों की। \n 3) तंबाकू की बिक्री। \n 4) तंबाकू शेल्फ की तस्वीर अटैच करें। \n 5) शॉप कीप टॉप सेलिंग ब्रांड्स से पूछें। \n 6) ऊर्जा पेय की बिक्री।',  
 'Accept': 'स्वीकार करना',  
 'Cancel': 'रद्द करना',  
 'Start New Survey': 'नया सर्वेक्षण शुरू करें',  
 'Last Survey Conducted on 26 Nov 2022':  
 'अंतिम सर्वेक्षण 26 नवंबर 2022 को आयोजित किया गया',  
 'Upload Invoices': 'चालान अपलोड करें',  
 'Last Invoice Uploaded on 26 Nov 2022':  
 'अंतिम चालान 26 नवंबर 2022 को अपलोड किया गया',  
 'Upload Stock Report': 'स्टॉक रिपोर्ट अपलोड करें',  
 'Last Report Uploaded on 26 Nov 2022':  
 'अंतिम रिपोर्ट 26 नवंबर 2022 को अपलोड की गई',  
 'Let’s Start...': 'चलो शुरू करो...',  
 'Quick Supermarket, Majan': 'क्विक सुपरमार्केट, माजन',  
 'Your slot has been blocked for\n60 min':  
 'आपका स्लॉट \n60 मिनट के लिए ब्लॉक कर दिया गया है',  
 'Start Survey Now!': 'अभी सर्वेक्षण शुरू करें!',  
 'Upload File': 'फ़ाइल अपलोड करें',  
 'Select File': 'फ़ाइल का चयन करें',  
 'Question': 'सवाल',  
 'Does the location have a POS system?':  
 'क्या स्थान में पीओएस सिस्टम है?',  
 'Total Numbers of cash counters?': 'कैश काउंटरों की कुल संख्या?',  
 'Do they sale tobacco?': 'क्या वे तंबाकू बेचते हैं?',  
 'Attach image of tobacco shelf.': 'तंबाकू शेल्फ की छवि संलग्न करें।',  
 'Ask shopkeeper top selling brands?':  
 'दुकानदार से पूछें सबसे ज्यादा बिकने वाले ब्रांड?',  
 'Take Picture': 'तस्वीर ले लो',  
 'Yes': 'हाँ',  
 'No': 'नहीं',  
 'Select Brand': 'ब्रांड का चयन करें',  
 'Nike': 'नाइके',  
 'Hermes': 'हेमीज़',  
 'Louis Vuitton': 'लुई वुइटन',  
 'Select Option': 'विकल्प चुनें',  
 'Select Amount': 'राशि का चयन करें',  
 'Upload Image': 'तस्विर अपलोड करना',  
 'Sign Out': 'साइन आउट',  
 'Language': 'भाषा',  
 'Preferences': 'पसंद',  
 'Rate the App': 'एप्लिकेशन की श्रेणी बताओ',  
 'Help': 'मदद',  
 'Support': 'सहायता',  
 'Rewards': 'पुरस्कार',  
 'You have 568 points': 'आपके 568 अंक हैं',  
 'Benefits': 'फ़ायदे',  
 'Notifications': 'सूचनाएं',  
 'Accounts': 'हिसाब किताब',  
 'Personal Information': 'व्यक्तिगत जानकारी',  
 'Your Account': 'आपका खाता',  
 'Profile': 'प्रोफ़ाइल',  
 'Date Of Birth': 'जन्म की तारीख',  
 'United Arab Emirates': 'संयुक्त अरब अमीरातs',  
 'Saved Payment Methods': 'सहेजे गए भुगतान के तरीके',  
 'Add New': 'नया जोड़ो',  
 'Add': 'जोड़ना',  
 'Add Payment Method': 'भुगतान विधि जोड़ें',  
 'First Abu Dhabi Bank': 'पहला अबू धाबी बैंक',  
 'Dubai Islamic Bank': 'दुबई इस्लामिक बैंक',  
 'Etisalat Wallet': 'एतिसलात वाले',  
 'Magnati': 'मैग्नेटी',  
 'Bank Account': 'बैंक खाता',  
 'E-Wallet': 'ई-बटुआ',  
 'Today\'s Notification': 'आज की अधिसूचना',  
 'Approved Review': 'स्वीकृत समीक्षा',  
 'Your Given Review about Shop is approved from Admin.':  
 'शॉप के बारे में आपकी दी गई समीक्षा एडमिन से स्वीकृत है।',  
 'Send via crypto network': 'क्रिप्टो नेटवर्क के माध्यम से भेजें',  
 'Send via email/ Phone/ Pay ID': 'ईमेल/फोन/पे आईडी द्वारा भेजें',  
 'send to a known crypto address\nvia crypto network':  
 'ज्ञात क्रिप्टो पते \n पर क्रिप्टो नेटवर्क के माध्यम से भेजें',  
 'Recipient must be a Algorthmi user':  
 'प्राप्तकर्ता एक Algorthmi उपयोगकर्ता होना चाहिए',  
 'Coins': 'सिक्के',  
 'Cash': 'नकद',  
 'You currently have': 'इस समय आपके पास',  
 'Link Account': 'खाते लिंक करें',  
 'Transfer to Bank/E-Wallet': 'बैंक / ई-वॉलेट में स्थानांतरण',  
 'Withdraw': 'निकालना',  
 'Transfer': 'स्थानांतरण',  
 'Withdrawal Money': 'निकासी धन',  
 'Al Ansari Exchange': 'अल अंसारी एक्सचेंज',  
 },  
 'ml\_IN': {  
 'Select your preferred language':  
 'നിങ്ങൾ ഇഷ്ടപ്പെടുന്ന ഭാഷ തിരഞ്ഞെടുക്കുക',  
 'Login': 'ലോഗിൻ',  
 'Enter Your Email': 'നിങ്ങളുടെ ഇമെയിൽ നൽകുക',  
 'Please enter a valid email': 'സാധുവായ ഒരു ഇമെയിൽ നൽകുക',  
 'Enter Your Password': 'നിങ്ങളുടെ പാസ്‌വേഡ് നൽകുക',  
 'Password not be empty!': 'പാസ്‌വേഡ് ശൂന്യമായിരിക്കരുത്!',  
 'Password must be greater than 5 character!':  
 'പാസ്‌വേഡ് 5 പ്രതീകത്തിൽ കൂടുതലായിരിക്കണം!',  
 'Forgot Password?': 'പാസ്വേഡ് മറന്നോ?',  
 'Forgot Password': 'പാസ്വേഡ് മറന്നോ',  
 'Don’t have an account ? ': 'ഒരു അക്കൗണ്ട് ഇല്ലേ? ',  
 'Sign Up': 'സൈൻ അപ്പ് ചെയ്യുക',  
 'Don’t worry! It happens. Please enter the email address associated with your account.':  
 'വിഷമിക്കേണ്ട! അത് സംഭവിക്കുന്നു. നിങ്ങളുടെ അക്കൗണ്ടുമായി ബന്ധപ്പെട്ട ഇമെയിൽ വിലാസം ദയവായി നൽകുക.',  
 'Submit': 'സമർപ്പിക്കുക',  
 'Reset Password': 'പാസ്‌വേഡ് \nപുനഃസജ്ജമാക്കുക',  
 'Your new password must be different from previous used passwords.':  
 'നിങ്ങളുടെ പുതിയ പാസ്‌വേഡ് മുമ്പ് ഉപയോഗിച്ച പാസ്‌വേഡുകളിൽ നിന്ന് വ്യത്യസ്തമായിരിക്കണം.',  
 'Password': 'പാസ്വേഡുകൾ',  
 'Confirm Password': 'പാസ്വേഡ് സ്ഥിരീകരിക്കുക',  
 'Select Role': 'റോൾ തിരഞ്ഞെടുക്കുക',  
 'Freelancers': 'ഫ്രീലാൻസർമാർ',  
 'Shop Owner': 'കട ഉടമ',  
 'Already have an account ? ': 'ഇതിനകം ഒരു അക്കൗണ്ട് ഉണ്ടോ? ',  
 'Upload Selfie': 'സെൽഫി അപ്‌ലോഡ്\nചെയ്യുക',  
 'Take Selfie for starting Questionnaire':  
 'ചോദ്യാവലി ആരംഭിക്കുന്നതിന് ഷോപ്പിന്റെ പേര് ലേബൽ സഹിതം സെൽഫി എടുക്കുക',  
 'Take Selfie': 'സെൽഫി എടുക്കുക',  
 'Take / Upload': 'എടുക്കുക / അപ്‌ലോഡ് ചെയ്യുക',  
 'Upload from Gallery': 'ഗാലറിയിൽ നിന്ന് അപ്‌ലോഡ് ചെയ്യുക',  
 'Next': 'അടുത്തത്',  
 'Scan Your ID': 'നിങ്ങളുടെ ഐഡി \nസ്കാൻ ചെയ്യുക',  
 'Personal Details\nConfirmation': 'വ്യക്തിഗത വിശദാംശങ്ങൾ സ്ഥിരീകരണം',  
 'Mobile & Email\nConfirmation': 'മൊബൈലും ഇമെയിലും\nസ്ഥിരീകരണം',  
 'Enter Verification\nCode': 'പരിശോധിച്ചുറപ്പിക്കൽ\nകോഡ് നൽകുക',  
 'Scan Your License': 'നിങ്ങളുടെ ലൈസൻസ് \nസ്കാൻ ചെയ്യുക',  
 'License Details\nConfirmation': 'ലൈസൻസ് വിശദാംശങ്ങൾ\nസ്ഥിരീകരണം',  
 'Confirm': 'സ്ഥിരീകരിക്കുക',  
 'Verify': 'സ്ഥിരീകരിക്കുക',  
 'Back': 'തിരികെ',  
 'Please confirm your\nPersonal Information':  
 'നിങ്ങളുടെ വ്യക്തിഗത \nവിവരങ്ങൾ സ്ഥിരീകരിക്കുക',  
 'ID Number': 'ഐഡി നമ്പർ',  
 'ID Number not be empty!': 'ഐഡി നമ്പർ ശൂന്യമായിരിക്കരുത്!',  
 'Full Name': 'പൂർണ്ണമായ പേര്',  
 'Full Name not be empty!': 'മുഴുവൻ പേര് ശൂന്യമല്ല!',  
 'Nationality': 'ദേശീയത',  
 'Nationality not be empty!': 'ദേശീയത ശൂന്യമാകരുത്!',  
 'Scan the back of your ': 'നിങ്ങളുടെ പിൻഭാഗം സ്കാൻ ചെയ്യുക ',  
 'Emirates ID': 'എമിറേറ്റ്സ് ഐഡി',  
 'Start Scanning': 'സ്കാനിംഗ് ആരംഭിക്കുക',  
 'Card Number': 'കാർഡ് നമ്പർ',  
 'Card Number not be empty!': 'കാർഡ് നമ്പർ ശൂന്യമായിരിക്കരുത്!',  
 'Please Place the back of your\nTrade License inside the frame':  
 'നിങ്ങളുടെ ട്രേഡ് ലൈസൻസിന്റെ \nപിൻഭാഗം ഫ്രെയിമിനുള്ളിൽ വയ്ക്കുക',  
 'Scan License': 'സ്കാൻ ലൈസൻസ്',  
 'Please Place the back of your\nEmirates ID inside the frame':  
 'നിങ്ങളുടെ എമിറേറ്റ്സ് ഐഡിയുടെ \nപിൻഭാഗം ഫ്രെയിമിനുള്ളിൽ വയ്ക്കുക',  
 'Scan Card': 'കാർഡ് സ്കാൻ ചെയ്യുക',  
 'Date of Birth': 'ജനനത്തീയതി',  
 'Date of Birth not be empty!': 'ജനനത്തീയതി ശൂന്യമല്ല!',  
 'Expiry Date': 'കാലഹരണപ്പെടുന്ന തീയതി',  
 'Expiry Date not be empty!': 'കാലഹരണ തീയതി ശൂന്യമല്ല!',  
 'Gender': 'ലിംഗഭേദം',  
 'Male': 'ആൺ',  
 'Female': 'സ്ത്രീ',  
 'Other': 'മറ്റുള്ളവ',  
 'Gender not be empty!': 'ലിംഗഭേദം ശൂന്യമല്ല!',  
 'Please provide your Mobile Phone Number and Email to continue':  
 'തുടരാൻ നിങ്ങളുടെ മൊബൈൽ ഫോൺ നമ്പറും ഇമെയിലും നൽകുക',  
 'Phone Number': 'ഫോൺ നമ്പർ',  
 'Email': 'ഇമെയിൽ',  
 'Please enter the OTP Code\nwe sent you':  
 'ദയവായി OTP കോഡ് നൽകുക\nഞങ്ങൾ നിങ്ങൾക്ക് അയച്ചു',  
 'I did not receive the OTP': 'എനിക്ക് OTP ലഭിച്ചില്ല',  
 'Resend Code': 'കോഡ് വീണ്ടും അയക്കുക',  
 'Please confirm your\nLicense Information':  
 'നിങ്ങളുടെ \nലൈസൻസ് വിവരങ്ങൾ സ്ഥിരീകരിക്കുക',  
 'License No': 'ലൈസൻസ് നമ്പർ',  
 'License No not be empty!': 'ലൈസൻസ് നമ്പർ ശൂന്യമാകരുത്!',  
 'Trade Name': 'വ്യാപാര നാമം',  
 'Trade Name not be empty!': 'വ്യാപാര നാമം ശൂന്യമായിരിക്കരുത്!',  
 'Scan the back of your\n': 'നിങ്ങളുടെ പിൻഭാഗം സ്കാൻ ചെയ്യുക\n',  
 'Trade License': 'വ്യാപാര ലൈസൻസ്',  
 'WellDone': 'നന്നായി ചെയ്തു',  
 'Congratulations': 'അഭിനന്ദനങ്ങൾ',  
 'You have successfully registered\nyour ALGORITHMI account':  
 'നിങ്ങളുടെ ALGORITHMI അക്കൗണ്ട് \n നിങ്ങൾ വിജയകരമായി രജിസ്റ്റർ ചെയ്തു',  
 'You have successfully added\nyour': 'നിങ്ങൾ \n നിങ്ങളുടെ',  
 'You\'re upload file and Submitted\nfor review':  
 'നിങ്ങൾ ഫയൽ അപ്‌ലോഡ് ചെയ്യുകയും \n അവലോകനത്തിനായി സമർപ്പിക്കുകയും ചെയ്തു',  
 'Your survey has been Finished and\nSubmitted for review':  
 'നിങ്ങളുടെ സർവേ പൂർത്തിയായി \n അവലോകനത്തിനായി സമർപ്പിച്ചു',  
 'Go to Dashboard': 'ഡാഷ്‌ബോർഡിലേക്ക് പോകുക',  
 'Dashboard': 'ഡാഷ്ബോർഡ്',  
 'Wallet': 'വാലറ്റ്',  
 'Settings': 'ക്രമീകരണം',  
 'Muhammad Usman': 'മുഹമ്മദ് ഉസ്മാൻ',  
 'Verified Account': 'സ്ഥിരീകരിച്ച അക്കൗണ്ട്',  
 'Current location: ': 'ഇപ്പോഴുള്ള സ്ഥലം: ',  
 'Al Barai, Dubai': 'അൽ ബറായ്, ദുബായ്',  
 'Account Type: ': 'അക്കൗണ്ട് തരം: ',  
 'Shop Keeper': 'ഷോപ്പ് കീപ്പർ',  
 'Member Status: ': 'അംഗ നില: ',  
 'Bronze': 'വെങ്കലം',  
 'Questioners: ': 'ചോദ്യകർത്താക്കൾ: ',  
 '3/5 Approved': '3/5 അംഗീകരിച്ചു',  
 'Questionnaire': 'ചോദ്യാവലി',  
 '1) Does the location have a POS system.\n2) # of counters.\n3) Sales of tobacco.\n4) Attach image of tobacco shelf.\n5) Ask shop keep top selling brands.\n6) Sales of energy drinks.':  
 '1) ലൊക്കേഷനിൽ POS സിസ്റ്റം ഉണ്ടോ. \n 2) # കൗണ്ടറുകൾ. \n 3) പുകയില വിൽപ്പന. \n 4) പുകയില ഷെൽഫിന്റെ ചിത്രം അറ്റാച്ചുചെയ്യുക. \n 5) ഏറ്റവും കൂടുതൽ വിറ്റഴിക്കപ്പെടുന്ന ബ്രാൻഡുകൾ സൂക്ഷിക്കാൻ ഷോപ്പിനോട് ആവശ്യപ്പെടുക. \n 6) ഊർജ്ജ പാനീയങ്ങളുടെ വിൽപ്പന.',  
 'Accept': 'സ്വീകരിക്കുക',  
 'Cancel': 'റദ്ദാക്കുക',  
 'Start New Survey': 'പുതിയ സർവേ ആരംഭിക്കുക',  
 'Last Survey Conducted on 26 Nov 2022':  
 '2022 നവംബർ 26-നാണ് അവസാനമായി സർവേ നടത്തിയത്',  
 'Upload Invoices': 'ഇൻവോയ്‌സുകൾ \nഅപ്‌ലോഡ് ചെയ്യുക',  
 'Last Invoice Uploaded on 26 Nov 2022':  
 '2022 നവംബർ 26-നാണ് അവസാനമായി ഇൻവോയ്സ് അപ്‌ലോഡ് ചെയ്തത്',  
 'Upload Stock Report': 'സ്റ്റോക്ക് റിപ്പോർട്ട് \nഅപ്ലോഡ് ചെയ്യുക',  
 'Last Report Uploaded on 26 Nov 2022':  
 '2022 നവംബർ 26-നാണ് അവസാനമായി റിപ്പോർട്ട് അപ്‌ലോഡ് ചെയ്തത്',  
 'Let’s Start...': 'നമുക്ക് തുടങ്ങാം...',  
 'Quick Supermarket, Majan': 'ദ്രുത സൂപ്പർമാർക്കറ്റ്, മജൻ',  
 'Your slot has been blocked for\n60 min':  
 'നിങ്ങളുടെ സ്ലോട്ട് \n60 മിനിറ്റായി തടഞ്ഞിരിക്കുന്നു',  
 'Start Survey Now!': 'ഇപ്പോൾ സർവേ ആരംഭിക്കുക!',  
 'Upload File': 'ഫയൽ അപ്‌ലോഡ് ചെയ്യുക',  
 'Select File': 'ഫയൽ തിരഞ്ഞെടുക്കുക',  
 'Question': 'ചോദ്യം',  
 'Does the location have a POS system?':  
 'ലൊക്കേഷനിൽ POS സിസ്റ്റം ഉണ്ടോ?',  
 'Total Numbers of cash counters?': 'ക്യാഷ് കൗണ്ടറുകളുടെ ആകെ എണ്ണം?',  
 'Do they sale tobacco?': 'അവർ പുകയില വിൽക്കുന്നുണ്ടോ?',  
 'Attach image of tobacco shelf.':  
 'പുകയില ഷെൽഫിന്റെ ചിത്രം അറ്റാച്ചുചെയ്യുക.',  
 'Ask shopkeeper top selling brands?':  
 'ഏറ്റവും കൂടുതൽ വിൽക്കുന്ന ബ്രാൻഡുകൾ കടയുടമയോട് ചോദിക്കണോ?',  
 'Take Picture': 'ചിത്രം എടുക്കുക',  
 'Yes': 'അതെ',  
 'No': 'ഇല്ല',  
 'Select Brand': 'ബ്രാൻഡ് തിരഞ്ഞെടുക്കുക',  
 'Nike': 'നൈക്ക്',  
 'Hermes': 'ഹെർമിസ്',  
 'Louis Vuitton': 'ലൂയി വിറ്റൺ',  
 'Select Option': 'ഓപ്ഷൻ തിരഞ്ഞെടുക്കുക',  
 'Select Amount': 'തുക തിരഞ്ഞെടുക്കുക',  
 'Upload Image': 'ചിത്രം അപ്ലോഡ് ചെയ്യുക',  
 'Sign Out': 'സൈൻ ഔട്ട്',  
 'Language': 'ഭാഷ',  
 'Preferences': 'മുൻഗണനകൾ',  
 'Rate the App': 'ആപ്പ് റേറ്റുചെയ്യുക',  
 'Help': 'സഹായം',  
 'Support': 'പിന്തുണ',  
 'Rewards': 'പ്രതിഫലം',  
 'You have 568 points': 'നിങ്ങൾക്ക് 568 പോയിന്റുണ്ട്',  
 'Benefits': 'ആനുകൂല്യങ്ങൾ',  
 'Notifications': 'അറിയിപ്പുകൾ',  
 'Accounts': 'അക്കൗണ്ടുകൾ',  
 'Personal Information': 'സ്വകാര്യ വിവരം',  
 'Your Account': 'നിങ്ങളുടെ അക്കൗണ്ട്',  
 'Profile': 'പ്രൊഫൈൽ',  
 'Date Of Birth': 'ജനനത്തീയതി',  
 'United Arab Emirates': 'യുണൈറ്റഡ് അറബ് എമിറേറ്റ്സ്',  
 'Saved Payment Methods': 'സംരക്ഷിച്ച പേയ്‌മെന്റ് രീതികൾ',  
 'Add New': 'പുതിയത് ചേർക്കുക',  
 'Add': 'ചേർക്കുക',  
 'Add Payment Method': 'പേയ്‌മെന്റ് രീതി ചേർക്കുക',  
 'First Abu Dhabi Bank': 'ആദ്യത്തെ അബുദാബി ബാങ്ക്',  
 'Dubai Islamic Bank': 'ദുബായ് ഇസ്ലാമിക് ബാങ്ക്',  
 'Etisalat Wallet': 'എത്തിസലാത്ത് വാലെ',  
 'Magnati': 'മാഗ്നതി',  
 'Bank Account': 'ബാങ്ക് അക്കൗണ്ട്',  
 'E-Wallet': 'ഇ-വാലറ്റ്',  
 'Today\'s Notification': 'ഇന്നത്തെ അറിയിപ്പ്',  
 'Approved Review': 'അംഗീകൃത അവലോകനം',  
 'Your Given Review about Shop is approved from Admin.':  
 'ഷോപ്പിനെ കുറിച്ചുള്ള നിങ്ങളുടെ അവലോകനം അഡ്മിനിൽ നിന്ന് അംഗീകരിക്കപ്പെട്ടിരിക്കുന്നു.',  
 'Send via crypto network': 'ക്രിപ്‌റ്റോ നെറ്റ്‌വർക്ക് വഴി അയയ്ക്കുക',  
 'Send via email/ Phone/ Pay ID': 'ഇമെയിൽ / ഫോൺ / പേ ഐഡി വഴി അയയ്ക്കുക',  
 'send to a known crypto address\nvia crypto network':  
 'ക്രിപ്‌റ്റോ നെറ്റ്‌വർക്ക് വഴി \n അറിയപ്പെടുന്ന ഒരു ക്രിപ്‌റ്റോ വിലാസത്തിലേക്ക് അയയ്‌ക്കുക',  
 'Recipient must be a Algorthmi user':  
 'സ്വീകർത്താവ് അൽഗോർത്മി ഉപയോക്താവായിരിക്കണം',  
 'Coins': 'നാണയങ്ങൾ',  
 'Cash': 'പണം',  
 'You currently have': 'നിങ്ങൾക്ക് നിലവിൽ ഉണ്ട്',  
 'Link Account': 'അക്കൗണ്ട് ലിങ്ക് ചെയ്യുക',  
 'Transfer to Bank/E-Wallet': 'ബാങ്ക് / ഇ-വാലറ്റിലേക്ക് ട്രാൻസ്ഫർ ചെയ്യുക',  
 'Withdraw': 'പിൻവലിക്കുക',  
 'Transfer': 'കൈമാറ്റം',  
 'Withdrawal Money': 'പണം പിൻവലിക്കൽ',  
 'Al Ansari Exchange': 'അൽ അൻസാരി എക്സ്ചേഞ്ച്',  
 },  
 'de\_DE': {  
 'Select your preferred language': 'Selecteer uw voorkeurstaal',  
 'Login': 'Anmeldung',  
 'Enter Your Email': 'Voer uw e-mailadres in',  
 'Please enter a valid email':  
 'Gelieve een geldig e-mailadres in te vullen',  
 'Enter Your Password': 'Voer uw wachtwoord in',  
 'Password not be empty!': 'Wachtwoord mag niet leeg zijn!',  
 'Password must be greater than 5 character!':  
 'Wachtwoord moet langer zijn dan 5 tekens!',  
 'Forgot Password?': 'Wachtwoord vergeten?',  
 'Forgot Password': 'Wachtwoord vergeten',  
 'Don’t have an account ? ': 'Heb je geen account? ',  
 'Sign Up': 'Aanmelden',  
 'Don’t worry! It happens. Please enter the email address associated with your account.':  
 'Maak je geen zorgen! Het gebeurt. Voer het e-mailadres in dat aan uw account is gekoppeld.',  
 'Submit': 'Indienen',  
 'Reset Password': 'Reset wachtwoord',  
 'Your new password must be different from previous used passwords.':  
 'Uw nieuwe wachtwoord moet verschillen van eerder gebruikte wachtwoorden.',  
 'Password': 'Wachtwoord',  
 'Confirm Password': 'Bevestig wachtwoord',  
 'Select Role': 'Rol selecteren',  
 'Freelancers': 'zzp\'ers',  
 'Shop Owner': 'Winkeleigenaar',  
 'Already have an account ? ': 'Heb je al een account? ',  
 'Upload Selfie': 'Selfie uploaden',  
 'Take Selfie for starting Questionnaire':  
 'Neem een selfie met een winkelnaamlabel om de vragenlijst te starten',  
 'Take Selfie': 'Maak een selfie',  
 'Take / Upload': 'Nemen / uploaden',  
 'Upload from Gallery': 'Uploaden vanuit Galerij',  
 'Next': 'Volgende',  
 'Scan Your ID': 'Scannen Sie \nIhren Ausweis',  
 'Personal Details\nConfirmation': 'Persönliche Daten\nBestätigung',  
 'Mobile & Email\nConfirmation': 'Handy & E-Mail\nBestätigung',  
 'Enter Verification\nCode': 'Bestätigungscode \neingeben',  
 'Scan Your License': 'Scannen Sie \nIhre Lizenz',  
 'License Details\nConfirmation': 'Lizenzdetails\nBestätigung',  
 'Confirm': 'Bestätigen',  
 'Verify': 'Verifizieren',  
 'Back': 'Zurück',  
 'Please confirm your\nPersonal Information':  
 'Bitte bestätigen Sie Ihre \n persönlichen Daten',  
 'ID Number': 'ID-Nummer',  
 'ID Number not be empty!': 'ID-Nummer darf nicht leer sein!',  
 'Full Name': 'Vollständiger Name',  
 'Full Name not be empty!': 'Vollständiger Name ist nicht leer!',  
 'Nationality': 'Staatsangehörigkeit',  
 'Nationality not be empty!': 'Nationalität nicht leer sein!',  
 'Scan the back of your ': 'Scannen Sie die Rückseite Ihres ',  
 'Emirates ID': 'Emirates-ID',  
 'Start Scanning': 'Scannen starten',  
 'Card Number': 'Kartennummer',  
 'Card Number not be empty!': 'Kartennummer darf nicht leer sein!',  
 'Please Place the back of your\nTrade License inside the frame':  
 'Bitte legen Sie die Rückseite \nIhres Gewerbescheins in den Rahmen',  
 'Scan License': 'Lizenz scannen',  
 'Please Place the back of your\nEmirates ID inside the frame':  
 'Bitte legen Sie die Rückseite Ihres\n Emirates-Ausweises in den Rahmen',  
 'Scan Card': 'Karte scannen',  
 'Date of Birth': 'Geburtsdatum',  
 'Date of Birth not be empty!': 'Geburtsdatum ist nicht leer!',  
 'Expiry Date': 'Verfallsdatum',  
 'Expiry Date not be empty!': 'Ablaufdatum ist nicht leer!',  
 'Gender': 'Geschlecht',  
 'Male': 'Männlich',  
 'Female': 'Weiblich',  
 'Other': 'Andere',  
 'Gender not be empty!': 'Geschlecht ist nicht leer!',  
 'Please provide your Mobile Phone Number and Email to continue':  
 'Bitte geben Sie Ihre Mobiltelefonnummer und E-Mail-Adresse an, um fortzufahren',  
 'Phone Number': 'Telefonnummer',  
 'Email': 'Email',  
 'Please enter the OTP Code\nwe sent you':  
 'Bitte geben Sie den OTP-Code ein,\nden wir Ihnen gesendet haben',  
 'I did not receive the OTP': 'Ich habe das OTP nicht erhalten',  
 'Resend Code': 'Code erneut senden',  
 'Please confirm your\nLicense Information':  
 'Bitte bestätigen Sie Ihre \nLizenzinformationen',  
 'License No': 'Lizenznummer',  
 'License No not be empty!': 'Lizenznummer darf nicht leer sein!',  
 'Trade Name': 'Handelsname',  
 'Trade Name not be empty!': 'Handelsname darf nicht leer sein!',  
 'Scan the back of your\n': 'Scannen Sie die Rückseite Ihres\n',  
 'Trade License': 'Handelslizenz',  
 'WellDone': 'Gut gemacht',  
 'Congratulations': 'Glückwunsch',  
 'You have successfully registered\nyour ALGORITHMI account':  
 'Sie haben Ihr ALGORITHMI-Konto \n erfolgreich registriert',  
 'You have successfully added\nyour':  
 'Sie haben \n Ihre erfolgreich hinzugefügt',  
 'You\'re upload file and Submitted\nfor review':  
 'Sie haben die Datei hochgeladen und \n zur Überprüfung eingereicht',  
 'Your survey has been Finished and\nSubmitted for review':  
 'Ihre Umfrage wurde abgeschlossen und \n zur Überprüfung eingereicht',  
 'Go to Dashboard': 'Gehen Sie zum Dashboard',  
 'Dashboard': 'Armaturenbrett',  
 'Wallet': 'Geldbörse',  
 'Settings': 'Einstellung',  
 'Muhammad Usman': 'Muhammad Usman',  
 'Verified Account': 'Verifiziertes Konto',  
 'Current location: ': 'Aktueller Standort: ',  
 'Al Barai, Dubai': 'Al-Barai, Dubai',  
 'Account Type: ': 'Konto Typ: ',  
 'Shop Keeper': 'Ladenbesitzer',  
 'Member Status: ': 'Mitgliedsstatus: ',  
 'Bronze': 'Bronze',  
 'Questioners: ': 'Fragesteller: ',  
 '3/5 Approved': '3/5 genehmigt',  
 'Questionnaire': 'Fragebogen',  
 '1) Does the location have a POS system.\n2) # of counters.\n3) Sales of tobacco.\n4) Attach image of tobacco shelf.\n5) Ask shop keep top selling brands.\n6) Sales of energy drinks.':  
 '1) Verfügt der Standort über ein POS-System? \n 2) Anzahl der Zähler. \n 3) Verkauf von Tabak. \n 4) Bild des Tabakregals anhängen. \n 5) Fragen Sie den Shop, um die meistverkauften Marken zu behalten. \n 6) Verkauf von Energydrinks.',  
 'Accept': 'Akzeptieren',  
 'Cancel': 'Stornieren',  
 'Start New Survey': 'Neue Umfrage starten',  
 'Last Survey Conducted on 26 Nov 2022':  
 'Letzte Umfrage durchgeführt am 26. November 2022',  
 'Upload Invoices': 'Rechnungen hochladen',  
 'Last Invoice Uploaded on 26 Nov 2022':  
 'Letzte Rechnung hochgeladen am 26. November 2022',  
 'Upload Stock Report': 'Bestandsbericht hochladen',  
 'Last Report Uploaded on 26 Nov 2022':  
 'Letzter Bericht hochgeladen am 26. November 2022',  
 'Let’s Start...': 'Lasst uns beginnen...',  
 'Quick Supermarket, Majan': 'Schneller Supermarkt, Majan',  
 'Your slot has been blocked for\n60 min':  
 'Ihr Slot wurde für \n60 Minuten gesperrt',  
 'Start Survey Now!': 'Jetzt Umfrage starten!',  
 'Upload File': 'Datei hochladen',  
 'Select File': 'Datei aussuchen',  
 'Question': 'Frage',  
 'Does the location have a POS system?':  
 'Verfügt der Standort über ein Kassensystem?',  
 'Total Numbers of cash counters?': 'Gesamtzahl der Kassenschalter?',  
 'Do they sale tobacco?': 'Verkaufen sie Tabak?',  
 'Attach image of tobacco shelf.': 'Bild des Tabakregals anhängen.',  
 'Ask shopkeeper top selling brands?':  
 'Fragen Sie den Ladenbesitzer nach den meistverkauften Marken?',  
 'Take Picture': 'Ein Bild machen',  
 'Yes': 'Ja',  
 'No': 'NEIN',  
 'Select Brand': 'Marke auswählen',  
 'Nike': 'Nike',  
 'Hermes': 'Hermes',  
 'Louis Vuitton': 'Louis Vuitton',  
 'Select Option': 'Wähle eine Option',  
 'Select Amount': 'Wählen Sie Betrag',  
 'Upload Image': 'Bild hochladen',  
 'Sign Out': 'Austragen',  
 'Language': 'Sprache',  
 'Preferences': 'Einstellungen',  
 'Rate the App': 'Bewerten Sie die App',  
 'Help': 'Hilfe',  
 'Support': 'Unterstützung',  
 'Rewards': 'Belohnung',  
 'You have 568 points': 'Sie haben 568 Punkte',  
 'Benefits': 'Vorteile',  
 'Notifications': 'Benachrichtigungen',  
 'Accounts': 'Konten',  
 'Personal Information': 'Persönliche Angaben',  
 'Your Account': 'Ihr Konto',  
 'Profile': 'Profil',  
 'Date Of Birth': 'Geburtsdatum',  
 'United Arab Emirates': 'Vereinigte Arabische Emirate',  
 'Saved Payment Methods': 'Gespeicherte Zahlungsmethoden',  
 'Add New': 'Neue hinzufügen',  
 'Add': 'Hinzufügen',  
 'Add Payment Method': 'Zahlungsmethode hinzufügen',  
 'First Abu Dhabi Bank': 'Erste Abu Dhabi Bank',  
 'Dubai Islamic Bank': 'Islamische Bank von Dubai',  
 'Etisalat Wallet': 'Etisalat Walle',  
 'Magnati': 'Magnati',  
 'Bank Account': 'Bankkonto',  
 'E-Wallet': 'E-Wallet',  
 'Today\'s Notification': 'Heutige Benachrichtigung',  
 'Approved Review': 'Genehmigte Bewertung',  
 'Your Given Review about Shop is approved from Admin.':  
 'Ihre abgegebene Bewertung über den Shop wurde vom Administrator genehmigt.',  
 'Send via crypto network': 'Über Krypto-Netzwerk senden',  
 'Send via email/ Phone/ Pay ID': 'Senden Sie per E-Mail / Telefon / Pay ID',  
 'send to a known crypto address\nvia crypto network':  
 'Senden an eine bekannte Krypto-Adresse \n über ein Krypto-Netzwerk',  
 'Recipient must be a Algorthmi user':  
 'Der Empfänger muss ein Algorthmi-Benutzer sein',  
 'Coins': 'Münzen',  
 'Cash': 'Kasse',  
 'You currently have': 'Derzeit hast due',  
 'Link Account': 'Konto verbinden',  
 'Transfer to Bank/E-Wallet': 'Überweisung an Bank / E-Wallet',  
 'Withdraw': 'Zurückziehen',  
 'Transfer': 'Überweisen',  
 'Withdrawal Money': 'Auszahlungsgeld',  
 'Al Ansari Exchange': 'Al-Ansari-Börse',  
 },  
 'ne\_IN': {  
 'Select your preferred language': 'आफ्नो मनपर्ने भाषा चयन गर्नुहोस्',  
 'Login': 'लग - इन',  
 'Enter Your Email': 'आफ्नो इमेल प्रविष्ट गर्नुहोस्',  
 'Please enter a valid email': 'कृपया मान्य इमेल प्रविष्ट गर्नुहोस्',  
 'Enter Your Password': 'आफ्नो पासवर्ड प्रविष्ट गर्नुहोस्',  
 'Password not be empty!': 'पासवर्ड खाली नहोस्!',  
 'Password must be greater than 5 character!':  
 'पासवर्ड 5 वर्ण भन्दा ठूलो हुनुपर्छ!',  
 'Forgot Password?': 'पासवर्ड भुल्नु भयो?',  
 'Forgot Password': 'पासवर्ड भुल्नु भयो',  
 'Don’t have an account ? ': 'खाता छैन? ',  
 'Sign Up': 'साइन अप',  
 'Don’t worry! It happens. Please enter the email address associated with your account.':  
 'चिन्ता नगर्नुहोस्! यस्तो हुन्छ। कृपया आफ्नो खातासँग सम्बन्धित इमेल ठेगाना प्रविष्ट गर्नुहोस्।',  
 'Submit': 'पेश गर्नुहोस्',  
 'Reset Password': 'पासवर्ड रिसेट',  
 'Your new password must be different from previous used passwords.':  
 'तपाईंको नयाँ पासवर्ड अघिल्लो प्रयोग गरिएका पासवर्डहरू भन्दा फरक हुनुपर्छ।',  
 'Password': 'पासवर्ड',  
 'Confirm Password': 'पासवर्ड सुनिश्चित गर्नुहोस',  
 'Select Role': 'भूमिका चयन गर्नुहोस्',  
 'Freelancers': 'फ्रीलान्सरहरू',  
 'Shop Owner': 'पसल मालिक',  
 'Already have an account ? ': 'पहिले नै खाता छ? ',  
 'Upload Selfie': 'सेल्फी अपलोड गर्नुहोस्',  
 'Take Selfie for starting Questionnaire':  
 'प्रश्नावली सुरु गर्नको लागि पसलको नाम लेबलको साथ सेल्फी लिनुहोस्',  
 'Take Selfie': 'सेल्फी लिनुहोस्',  
 'Take / Upload': 'लिनुहोस् / अपलोड गर्नुहोस्',  
 'Upload from Gallery': 'ग्यालरीबाट अपलोड गर्नुहोस्',  
 'Next': 'अर्को',  
 'Scan Your ID': 'आफ्नो आईडी \nस्क्यान गर्नुहोस्',  
 'Personal Details\nConfirmation': 'व्यक्तिगत विवरणहरू\nपुष्टि',  
 'Mobile & Email\nConfirmation': 'मोबाइल र इमेल\nपुष्टि',  
 'Enter Verification\nCode': 'प्रमाणीकरण\nकोड प्रविष्ट गर्नुहोस्',  
 'Scan Your License': 'तपाईंको लाइसेन्स \nस्क्यान गर्नुहोस्',  
 'License Details\nConfirmation': 'इजाजतपत्र विवरणहरू\nपुष्टि',  
 'Confirm': 'पुष्टि गर्नुहोस्',  
 'Verify': 'प्रमाणित गर्नुहोस्',  
 'Back': 'पछाडि',  
 'Please confirm your\nPersonal Information':  
 'कृपया आफ्नो \n व्यक्तिगत जानकारी पुष्टि गर्नुहोस्',  
 'ID Number': 'आईडी नम्बर',  
 'ID Number not be empty!': 'आईडी नम्बर खाली छैन!',  
 'Full Name': 'पुरा नाम',  
 'Full Name not be empty!': 'पूरा नाम खाली छैन!',  
 'Nationality': 'राष्ट्रियता',  
 'Nationality not be empty!': 'राष्ट्रियता खाली छैन!',  
 'Scan the back of your ': 'आफ्नो पछाडि स्क्यान गर्नुहोस् ',  
 'Emirates ID': 'एमिरेट्स आईडी',  
 'Start Scanning': 'स्क्यान सुरु गर्नुहोस्',  
 'Card Number': 'कार्ड नम्बर',  
 'Card Number not be empty!': 'कार्ड नम्बर खाली छैन!',  
 'Please Place the back of your\nTrade License inside the frame':  
 'कृपया आफ्नो ट्रेड लाइसेन्सको \nपछाडि फ्रेम भित्र राख्नुहोस्',  
 'Scan License': 'स्क्यान लाइसेन्स',  
 'Please Place the back of your\nEmirates ID inside the frame':  
 'कृपया आफ्नो एमिरेट्स आईडीको\n पछाडि फ्रेम भित्र राख्नुहोस्',  
 'Scan Card': 'स्क्यान कार्ड',  
 'Date of Birth': 'जन्म मिति',  
 'Date of Birth not be empty!': 'जन्म मिति खाली छैन!',  
 'Expiry Date': 'म्याद सकिने मिति',  
 'Expiry Date not be empty!': 'म्याद सकिने मिति खाली छैन!',  
 'Gender': 'लिङ्ग',  
 'Male': 'पुरुष',  
 'Female': 'महिला',  
 'Other': 'अन्य',  
 'Gender not be empty!': 'लिङ्ग खाली छैन!',  
 'Please provide your Mobile Phone Number and Email to continue':  
 'जारी राख्न कृपया आफ्नो मोबाइल फोन नम्बर र इमेल प्रदान गर्नुहोस्',  
 'Phone Number': 'फोन नम्बर',  
 'Email': 'इमेल',  
 'Please enter the OTP Code\nwe sent you':  
 'कृपया OTP कोड प्रविष्ट गर्नुहोस्\nहामीले तपाईंलाई पठाउनुभएको छ',  
 'I did not receive the OTP': 'मैले OTP प्राप्त गरेन',  
 'Resend Code': 'कोड पुन: पठाउनुहोस्',  
 'Please confirm your\nLicense Information':  
 'कृपया आफ्नो \nलाइसेन्स जानकारी पुष्टि गर्नुहोस्',  
 'License No': 'इजाजतपत्र नं',  
 'License No not be empty!': 'लाइसेन्स खाली छैन!',  
 'Trade Name': 'व्यापार नाम',  
 'Trade Name not be empty!': 'व्यापार नाम खाली हुनु हुँदैन!',  
 'Scan the back of your\n': 'आफ्नो पछाडि स्क्यान गर्नुहोस्\n',  
 'Trade License': 'ट्रेड लाइसेन्स',  
 'WellDone': 'राम्रो काम',  
 'Congratulations': 'बधाई छ',  
 'You have successfully registered\nyour ALGORITHMI account':  
 'तपाईंले सफलतापूर्वक आफ्नो ALGORITHMI खाता \n दर्ता गर्नुभएको छ',  
 'You have successfully added\nyour':  
 'तपाईंले सफलतापूर्वक थप्नुभएको छ \n तपाईंको',  
 'You\'re upload file and Submitted\nfor review':  
 'तपाईंले फाइल अपलोड गर्नुभयो र समीक्षाको लागि \n पेस गर्नुभयो',  
 'Your survey has been Finished and\nSubmitted for review':  
 'तपाईंको सर्वेक्षण समाप्त भयो र \n समीक्षाको लागि पेश गरियो',  
 'Go to Dashboard': 'ड्यासबोर्डमा जानुहोस्',  
 'Dashboard': 'ड्यासबोर्ड',  
 'Wallet': 'वालेट',  
 'Settings': 'सेटिङ',  
 'Muhammad Usman': 'मोहम्मद उस्मान',  
 'Verified Account': 'प्रमाणित खाता',  
 'Current location: ': 'वर्तमान स्थान: ',  
 'Al Barai, Dubai': 'अल बाराई, दुबई',  
 'Account Type: ': 'खाताको प्रकार: ',  
 'Shop Keeper': 'पसल किपर',  
 'Member Status: ': 'सदस्य स्थिति: ',  
 'Bronze': 'कांस्य',  
 'Questioners: ': 'प्रश्नकर्ताहरू: ',  
 '3/5 Approved': '3/5 स्वीकृत',  
 'Questionnaire': 'प्रश्नावली',  
 '1) Does the location have a POS system.\n2) # of counters.\n3) Sales of tobacco.\n4) Attach image of tobacco shelf.\n5) Ask shop keep top selling brands.\n6) Sales of energy drinks.':  
 '1) के स्थानमा POS प्रणाली छ। \n २) # काउन्टरहरू। \n 3) सुर्तीको बिक्री। \n 4) तंबाकू शेल्फको छवि संलग्न गर्नुहोस्। \n 5) पसललाई शीर्ष बिक्री हुने ब्रान्डहरू राख्न सोध्नुहोस्। \n ६) ऊर्जा पेयको बिक्री।',  
 'Accept': 'स्वीकार गर्नुहोस्',  
 'Cancel': 'रद्द गर्नुहोस्',  
 'Start New Survey': 'नयाँ सर्वेक्षण सुरु गर्नुहोस्',  
 'Last Survey Conducted on 26 Nov 2022':  
 'पछिल्लो सर्वेक्षण २६ नोभेम्बर २०२२ मा गरिएको थियो',  
 'Upload Invoices': 'इनभ्वाइसहरू अपलोड गर्नुहोस्',  
 'Last Invoice Uploaded on 26 Nov 2022':  
 'अन्तिम इनभ्वाइस २६ नोभेम्बर २०२२ मा अपलोड गरिएको थियो',  
 'Upload Stock Report': 'स्टक रिपोर्ट अपलोड गर्नुहोस्',  
 'Last Report Uploaded on 26 Nov 2022':  
 'पछिल्लो रिपोर्ट २६ नोभेम्बर २०२२ मा अपलोड गरिएको थियो',  
 'Let’s Start...': 'सुरु गरौँ...',  
 'Quick Supermarket, Majan': 'द्रुत सुपरमार्केट, माजन',  
 'Your slot has been blocked for\n60 min':  
 'तपाईंको स्लट \n60 मिनेटको लागि अवरुद्ध गरिएको छ',  
 'Start Survey Now!': 'अब सर्वेक्षण सुरु गर्नुहोस्!',  
 'Upload File': 'फाइल अपलोड गर्नुहोस्',  
 'Select File': 'फाइल चयन गर्नुहोस्',  
 'Question': 'प्रश्न',  
 'Does the location have a POS system?': 'के स्थानमा POS प्रणाली छ?',  
 'Total Numbers of cash counters?': 'नगद काउन्टरहरूको कुल संख्या?',  
 'Do they sale tobacco?': 'के तिनीहरू सुर्ती बेच्छन्?',  
 'Attach image of tobacco shelf.':  
 'तंबाकू शेल्फको छवि संलग्न गर्नुहोस्।',  
 'Ask shopkeeper top selling brands?':  
 'पसललाई सबैभन्दा बढी बिक्री हुने ब्रान्डहरू सोध्नुहोस्?',  
 'Take Picture': 'तस्वीर लिनुहोस्',  
 'Yes': 'हो',  
 'No': 'छैन',  
 'Select Brand': 'ब्रान्ड चयन गर्नुहोस्',  
 'Nike': 'नाइके',  
 'Hermes': 'हर्मिस',  
 'Louis Vuitton': 'लुइसभिटन',  
 'Select Option': 'विकल्प चयन गर्नुहोस्',  
 'Select Amount': 'रकम चयन गर्नुहोस्',  
 'Upload Image': 'छवि अपलोड गर्नुहोस्',  
 'Sign Out': 'साइन आउट गर्नुहोस्',  
 'Language': 'भाषा',  
 'Preferences': 'प्राथमिकताहरू',  
 'Rate the App': 'एप मूल्याङ्कन गर्नुहोस्',  
 'Help': 'मद्दत गर्नुहोस्',  
 'Support': 'समर्थन',  
 'Rewards': 'पुरस्कारहरू',  
 'You have 568 points': 'तपाईंसँग 568 अंक छ',  
 'Benefits': 'फाइदाहरू',  
 'Notifications': 'सूचनाहरू',  
 'Accounts': 'खाताहरू',  
 'Personal Information': 'व्यक्तिगत जानकारी',  
 'Your Account': 'तपाईंको खाता',  
 'Profile': 'प्रोफाइल',  
 'Date Of Birth': 'जन्म मिति',  
 'United Arab Emirates': 'संयुक्त अरब इमिरेट्स',  
 'Saved Payment Methods': 'सुरक्षित भुक्तानी विधिहरू',  
 'Add New': 'नयाँ थप्नुहोस्',  
 'Add': 'थप्नुहोस्',  
 'Add Payment Method': 'भुक्तानी विधि थप्नुहोस्',  
 'First Abu Dhabi Bank': 'पहिलो अबु धाबी बैंक',  
 'Dubai Islamic Bank': 'दुबई इस्लामिक बैंक',  
 'Etisalat Wallet': 'एतिसलात वाले',  
 'Magnati': 'Magnati',  
 'Bank Account': 'बैंक खाता',  
 'E-Wallet': 'ई-वालेट',  
 'Today\'s Notification': 'आजको सूचना',  
 'Approved Review': 'स्वीकृत समीक्षा',  
 'Your Given Review about Shop is approved from Admin.':  
 'पसलको बारेमा तपाईंको दिईएको समीक्षा प्रशासकबाट अनुमोदन गरिएको छ।',  
 'Send via crypto network': 'क्रिप्टो नेटवर्क मार्फत पठाउनुहोस्',  
 'Send via email/ Phone/ Pay ID': 'इमेल/फोन/भुक्तानी आईडी मार्फत पठाउनुहोस्',  
 'send to a known crypto address\nvia crypto network':  
 'क्रिप्टो नेटवर्क मार्फत \n ज्ञात क्रिप्टो ठेगानामा पठाउनुहोस्',  
 'Recipient must be a Algorthmi user':  
 'प्रापक एल्गोरथमी प्रयोगकर्ता हुनुपर्छ',  
 'Coins': 'सिक्का',  
 'Cash': 'नगद',  
 'You currently have': 'तपाईंसँग हाल छ',  
 'Link Account': 'खाता लिङ्क गर्नुहोस्',  
 'Transfer to Bank/E-Wallet': 'बैंक / ई-वालेटमा स्थानान्तरण गर्नुहोस्',  
 'Withdraw': 'निकाल्नु',  
 'Transfer': 'स्थानान्तरण',  
 'Withdrawal Money': 'निकासी पैसा',  
 'Al Ansari Exchange': 'अल अन्सारी एक्सचेन्ज',  
 },  
 'bn\_BD': {  
 'Select your preferred language': 'আপনার পছন্দের ভাষা নির্বাচন করুন',  
 'Login': 'প্রবেশ করুন',  
 'Enter Your Email': 'তুমার ইমেইল প্রবেশ করাও',  
 'Please enter a valid email': 'একটি বৈধ ইমেইল প্রবেশ করুন',  
 'Enter Your Password': 'আপনার পাসওয়ার্ড লিখুন',  
 'Password not be empty!': 'পাসওয়ার্ড খালি থাকবে না!',  
 'Password must be greater than 5 character!':  
 'পাসওয়ার্ড 5 অক্ষরের বেশি হতে হবে!',  
 'Forgot Password?': 'পাসওয়ার্ড ভুলে গেছেন?',  
 'Forgot Password': 'পাসওয়ার্ড ভুলে গেছেন',  
 'Don’t have an account ? ': 'একটি অ্যাকাউন্ট নেই? ',  
 'Sign Up': 'নিবন্ধন করুন',  
 'Don’t worry! It happens. Please enter the email address associated with your account.':  
 'চিন্তা করবেন না! এটা ঘটে। আপনার অ্যাকাউন্টের সাথে যুক্ত ইমেইল ঠিকানা লিখুন।',  
 'Submit': 'জমা দিন',  
 'Reset Password': 'পাসওয়ার্ড রিসেট করুন',  
 'Your new password must be different from previous used passwords.':  
 'আপনার নতুন পাসওয়ার্ড আগের ব্যবহৃত পাসওয়ার্ড থেকে আলাদা হতে হবে।',  
 'Password': 'পাসওয়ার্ড',  
 'Confirm Password': 'পাসওয়ার্ড নিশ্চিত করুন',  
 'Select Role': 'ভূমিকা নির্বাচন করুন',  
 'Freelancers': 'ফ্রিল্যান্সার',  
 'Shop Owner': 'দোকান মালিক',  
 'Already have an account ? ': 'ইতিমধ্যে একটি সদস্যপদ আছে? ',  
 'Upload Selfie': 'সেলফি আপলোড করুন',  
 'Take Selfie for starting Questionnaire':  
 'প্রশ্নাবলী শুরু করার জন্য দোকানের নামের লেবেল সহ সেলফি নিন',  
 'Take Selfie': 'সেলফি তুলুন',  
 'Take / Upload': 'নিন/আপলোড করুন',  
 'Upload from Gallery': 'গ্যালারি থেকে আপলোড করুন',  
 'Next': 'পরবর্তী',  
 'Scan Your ID': 'আপনার আইডি \nস্ক্যান করুন',  
 'Personal Details\nConfirmation': 'ব্যক্তিগত বিবরণ\nনিশ্চিতকরণ',  
 'Mobile & Email\nConfirmation': 'মোবাইল এবং ইমেল\nনিশ্চিতকরণ',  
 'Enter Verification\nCode': 'যাচাইকরণ\nকোড লিখুন',  
 'Scan Your License': 'আপনার লাইসেন্স \nস্ক্যান করুন',  
 'License Details\nConfirmation': 'লাইসেন্সের বিবরণ\nনিশ্চিতকরণ',  
 'Confirm': 'নিশ্চিত করুন',  
 'Verify': 'যাচাই করুন',  
 'Back': 'পেছনে',  
 'Please confirm your\nPersonal Information':  
 'অনুগ্রহ করে আপনার \n ব্যক্তিগত তথ্য নিশ্চিত করুন',  
 'ID Number': 'আইডি নাম্বার',  
 'ID Number not be empty!': 'আইডি নম্বর খালি থাকবে না!',  
 'Full Name': 'পুরো নাম',  
 'Full Name not be empty!': 'পুরো নাম খালি নেই!',  
 'Nationality': 'জাতীয়তা',  
 'Nationality not be empty!': 'জাতীয়তা খালি হবে না!',  
 'Scan the back of your ': 'আপনার পিছনে স্ক্যান করুন ',  
 'Emirates ID': 'এমিরেটস আইডি',  
 'Start Scanning': 'স্ক্যান করা শুরু করুন',  
 'Card Number': 'কার্ড নম্বর',  
 'Card Number not be empty!': 'কার্ড নম্বর খালি হবে না!',  
 'Please Place the back of your\nTrade License inside the frame':  
 'অনুগ্রহ করে আপনার ট্রেড লাইসেন্সের \nপিছনে ফ্রেমের ভিতরে রাখুন',  
 'Scan License': 'স্ক্যান লাইসেন্স',  
 'Please Place the back of your\nEmirates ID inside the frame':  
 'অনুগ্রহ করে আপনার এমিরেটস \nআইডির পিছনে ফ্রেমের ভিতরে রাখুন',  
 'Scan Card': 'স্ক্যান কার্ড',  
 'Date of Birth': 'জন্ম তারিখ',  
 'Date of Birth not be empty!': 'জন্ম তারিখ খালি নেই!',  
 'Expiry Date': 'মেয়াদ শেষ হওয়ার তারিখ',  
 'Expiry Date not be empty!': 'মেয়াদ শেষ হওয়ার তারিখ খালি নেই!',  
 'Gender': 'লিঙ্গ',  
 'Male': 'পুরুষ',  
 'Female': 'মহিলা',  
 'Other': 'অন্যান্য',  
 'Gender not be empty!': 'লিঙ্গ খালি নয়!',  
 'Please provide your Mobile Phone Number and Email to continue':  
 'চালিয়ে যেতে আপনার মোবাইল ফোন নম্বর এবং ইমেল প্রদান করুন',  
 'Phone Number': 'ফোন নম্বর',  
 'Email': 'ইমেইল',  
 'Please enter the OTP Code\nwe sent you':  
 'অনুগ্রহ করে OTP কোড লিখুন\nআমরা আপনাকে পাঠিয়েছি',  
 'I did not receive the OTP': 'আমি ওটিপি পাইনি',  
 'Resend Code': 'পুনরায় পাঠানো কোড',  
 'Please confirm your\nLicense Information':  
 'অনুগ্রহ করে আপনার \nলাইসেন্স তথ্য নিশ্চিত করুন',  
 'License No': 'লাইসেন্স নং',  
 'License No not be empty!': 'লাইসেন্স খালি হবে না!',  
 'Trade Name': 'বাণিজ্যিক নাম',  
 'Trade Name not be empty!': 'ট্রেডের নাম খালি হবে না!',  
 'Scan the back of your\n': 'আপনার পিছনে স্ক্যান করুন',  
 'Trade License': 'ট্রেড লাইসেন্স',  
 'WellDone': 'সাবাশ',  
 'Congratulations': 'অভিনন্দন',  
 'You have successfully registered\nyour ALGORITHMI account':  
 'আপনি সফলভাবে আপনার ALGORITHMI অ্যাকাউন্ট \n নিবন্ধন করেছেন৷',  
 'You have successfully added\nyour':  
 'আপনি সফলভাবে যোগ করেছেন \n আপনার',  
 'You\'re upload file and Submitted\nfor review':  
 'আপনি ফাইল আপলোড করছেন এবং পর্যালোচনার জন্য \n জমা দিয়েছেন',  
 'Your survey has been Finished and\nSubmitted for review':  
 'আপনার সমীক্ষা শেষ হয়েছে এবং \n পর্যালোচনার জন্য জমা দেওয়া হয়েছে৷',  
 'Go to Dashboard': 'ড্যাশবোর্ড এ যান',  
 'Dashboard': 'ড্যাশবোর্ড',  
 'Wallet': 'ওয়ালেট',  
 'Settings': 'বিন্যাস',  
 'Muhammad Usman': 'মুহাম্মদ উসমান',  
 'Verified Account': 'যাচাইকৃত একাউন্ট',  
 'Current location: ': 'এখন যেখানে আছ: ',  
 'Al Barai, Dubai': 'আল বারাই, দুবাই',  
 'Account Type: ': 'অ্যাকাউন্ট ধরন: ',  
 'Shop Keeper': 'দোকান কিপার',  
 'Member Status: ': 'সদস্যের অবস্থা: ',  
 'Bronze': 'ব্রোঞ্জ',  
 'Questioners: ': 'প্রশ্নকর্তা: ',  
 '3/5 Approved': '3/5 অনুমোদিত',  
 'Questionnaire': 'প্রশ্নপত্র',  
 '1) Does the location have a POS system.\n2) # of counters.\n3) Sales of tobacco.\n4) Attach image of tobacco shelf.\n5) Ask shop keep top selling brands.\n6) Sales of energy drinks.':  
 '1) অবস্থানের একটি POS সিস্টেম আছে কি? \n 2) # কাউন্টার। \n 3) তামাক বিক্রি। \n 4) তামাকের শেলফের ছবি সংযুক্ত করুন। \n 5) দোকানে সবচেয়ে বেশি বিক্রি হওয়া ব্র্যান্ড রাখতে বলুন। \n 6) শক্তি পানীয় বিক্রি।',  
 'Accept': 'গ্রহণ করুন',  
 'Cancel': 'বাতিল করুন',  
 'Start New Survey': 'নতুন জরিপ শুরু করুন',  
 'Last Survey Conducted on 26 Nov 2022':  
 '26 নভেম্বর 2022-এ শেষ সমীক্ষা করা হয়েছিল',  
 'Upload Invoices': 'চালান আপলোড করুন',  
 'Last Invoice Uploaded on 26 Nov 2022':  
 '26 নভেম্বর 2022-এ শেষ চালান আপলোড করা হয়েছে',  
 'Upload Stock Report': 'আপলোড স্টক রিপোর্ট',  
 'Last Report Uploaded on 26 Nov 2022':  
 '26 নভেম্বর 2022-এ শেষ রিপোর্ট আপলোড করা হয়েছে',  
 'Let’s Start...': 'চল শুরু করি...',  
 'Quick Supermarket, Majan': 'কুইক সুপার মার্কেট, মাজন',  
 'Your slot has been blocked for\n60 min':  
 'আপনার স্লট \n60 মিনিটের জন্য ব্লক করা হয়েছে',  
 'Start Survey Now!': 'এখনই জরিপ শুরু করুন!',  
 'Upload File': 'ফাইল আপলোড করুন',  
 'Select File': 'নথি নির্বাচন',  
 'Question': 'প্রশ্ন',  
 'Does the location have a POS system?':  
 'অবস্থান একটি POS সিস্টেম আছে?',  
 'Total Numbers of cash counters?': 'ক্যাশ কাউন্টার মোট সংখ্যা?',  
 'Do they sale tobacco?': 'তারা কি তামাক বিক্রি করে?',  
 'Attach image of tobacco shelf.': 'তামাকের শেলফের ছবি সংযুক্ত করুন।',  
 'Ask shopkeeper top selling brands?':  
 'দোকানদারকে জিজ্ঞাসা করুন শীর্ষ বিক্রিত ব্র্যান্ড?',  
 'Take Picture': 'ছবি তোলা',  
 'Yes': 'হ্যাঁ',  
 'No': 'না',  
 'Select Brand': 'ব্র্যান্ড নির্বাচন করুন',  
 'Nike': 'নাইকি',  
 'Hermes': 'হার্মিস',  
 'Louis Vuitton': 'লুই ভিটন',  
 'Select Option': 'বিকল্প নির্বাচন',  
 'Select Amount': 'পরিমাণ নির্বাচন করুন',  
 'Upload Image': 'চিত্র আপলোড',  
 'Sign Out': 'সাইন আউট',  
 'Language': 'ভাষা',  
 'Preferences': 'পছন্দসমূহ',  
 'Rate the App': 'অ্যাপ্লিকেশন রেট',  
 'Help': 'সাহায্য',  
 'Support': 'সমর্থন',  
 'Rewards': 'পুরস্কার',  
 'You have 568 points': 'আপনার 568 পয়েন্ট আছে',  
 'Benefits': 'সুবিধা',  
 'Notifications': 'বিজ্ঞপ্তি',  
 'Accounts': 'হিসাব',  
 'Personal Information': 'ব্যক্তিগত তথ্য',  
 'Your Account': 'আপনার অ্যাকাউন্ট',  
 'Profile': 'প্রোফাইল',  
 'Date Of Birth': 'জন্ম তারিখ',  
 'United Arab Emirates': 'সংযুক্ত আরব আমিরাত',  
 'Saved Payment Methods': 'সংরক্ষিত পেমেন্ট পদ্ধতি',  
 'Add New': 'নতুন যোগ করুন',  
 'Add': 'যোগ করুন',  
 'Add Payment Method': 'পেমেন্ট পদ্ধতি যোগ করুন',  
 'First Abu Dhabi Bank': 'প্রথম আবুধাবি ব্যাংক',  
 'Dubai Islamic Bank': 'দুবাই ইসলামিক ব্যাংক',  
 'Etisalat Wallet': 'এতিসালাত ওয়ালে',  
 'Magnati': 'মাগনাতি',  
 'Bank Account': 'ব্যাংক হিসাব',  
 'E-Wallet': 'ই-ওয়ালেট',  
 'Today\'s Notification': 'আজকের বিজ্ঞপ্তি',  
 'Approved Review': 'অনুমোদিত পর্যালোচনা',  
 'Your Given Review about Shop is approved from Admin.':  
 'দোকান সম্পর্কে আপনার দেওয়া পর্যালোচনা অ্যাডমিনের কাছ থেকে অনুমোদিত হয়েছে।',  
 'Send via crypto network': 'ক্রিপ্টো নেটওয়ার্কের মাধ্যমে পাঠান',  
 'Send via email/ Phone/ Pay ID': 'ইমেইল/ফোন/পে আইডির মাধ্যমে পাঠান',  
 'send to a known crypto address\nvia crypto network':  
 'একটি পরিচিত ক্রিপ্টো ঠিকানায় পাঠান \n ক্রিপ্টো নেটওয়ার্কের মাধ্যমে',  
 'Recipient must be a Algorthmi user':  
 'প্রাপক অবশ্যই অ্যালগর্থমি ব্যবহারকারী হতে হবে',  
 'Coins': 'কয়েন',  
 'Cash': 'নগদ',  
 'You currently have': 'আপনি বর্তমানে আছে',  
 'Link Account': 'অ্যাকাউন্ট লিঙ্ক করুন',  
 'Transfer to Bank/E-Wallet': 'ব্যাংক/ই-ওয়ালেটে স্থানান্তর করুন',  
 'Withdraw': 'প্রত্যাহার করুন',  
 'Transfer': 'স্থানান্তর',  
 'Withdrawal Money': 'টাকা তোলা',  
 'Al Ansari Exchange': 'আল আনসারী এক্সচেঞ্জ',  
 },  
 'ur\_PK': {  
 'Select your preferred language': 'اپنی پسند کی زبان منتخب کریں۔',  
 'Login': 'لاگ ان کریں',  
 'Enter Your Email': 'اپنا ای میل درج کریں۔',  
 'Please enter a valid email': 'ایک درست ای میل درج کریں',  
 'Enter Your Password': 'اپنا پاس ورڈ درج کریں',  
 'Password not be empty!': 'پاس ورڈ خالی نہ ہو!',  
 'Password must be greater than 5 character!':  
 'پاس ورڈ 5 حروف سے بڑا ہونا چاہیے!',  
 'Forgot Password?': 'پاسورڈ بھول گے؟',  
 'Forgot Password': 'پاسورڈ بھول گے',  
 'Don’t have an account ? ': 'اکاؤنٹ نہیں ہے؟ ',  
 'Sign Up': 'سائن اپ',  
 'Don’t worry! It happens. Please enter the email address associated with your account.':  
 'فکر مت کرو! یہ ہوتا ہے. براہ کرم اپنے اکاؤنٹ سے وابستہ ای میل ایڈریس درج کریں۔',  
 'Submit': 'جمع کرائیں',  
 'Reset Password': 'پاس ورڈ ری سیٹ',  
 'Your new password must be different from previous used passwords.':  
 'آپ کا نیا پاس ورڈ پہلے استعمال شدہ پاس ورڈز سے مختلف ہونا چاہیے۔',  
 'Password': 'پاس ورڈ',  
 'Confirm Password': 'پاس ورڈ کی تصدیق کریں۔',  
 'Select Role': 'کردار منتخب کریں۔',  
 'Freelancers': 'فری لانسرز',  
 'Shop Owner': 'دکان کے مالک',  
 'Already have an account ? ': 'پہلے سے ہی اکاؤنٹ ہے؟ ',  
 'Upload Selfie': 'سیلفی اپ لوڈ کریں',  
 'Take Selfie for starting Questionnaire':  
 'سوالنامہ شروع کرنے کے لیے دکان کے نام کے لیبل کے ساتھ سیلفی لیں۔',  
 'Take Selfie': 'سیلفی لیں۔',  
 'Take / Upload': 'لیں / اپ لوڈ کریں۔',  
 'Upload from Gallery': 'گیلری سے اپ لوڈ کریں۔',  
 'Next': 'اگلے',  
 'Scan Your ID': 'اپنی آئی ڈی \nاسکین کریں',  
 'Personal Details\nConfirmation': 'ذاتی تفصیلات\nتصدیق',  
 'Mobile & Email\nConfirmation': 'موبائل اور ای میل\nتصدیق',  
 'Enter Verification\nCode': 'تصدیقی کوڈ \nداخل کریں',  
 'Scan Your License': 'اپنا لائسنس\nاسکین کریں',  
 'License Details\nConfirmation': 'لائسنس کی تفصیلات\nتصدیق',  
 'Confirm': 'تصدیق کریں۔',  
 'Verify': 'تصدیق کریں۔',  
 'Back': 'پیچھے',  
 'Please confirm your\nPersonal Information':  
 'براہ کرم اپنی ذاتی\n معلومات کی تصدیق کریں',  
 'ID Number': 'شناختی نمبر',  
 'ID Number not be empty!': 'شناختی نمبر خالی نہ ہو!',  
 'Full Name': 'پورا نام',  
 'Full Name not be empty!': 'پورا نام خالی نہیں ہے!',  
 'Nationality': 'قومیت',  
 'Nationality not be empty!': 'قومیت خالی نہ ہو!',  
 'Scan the back of your ': 'اپنے پیچھے کو اسکین کریں۔ ',  
 'Emirates ID': 'ایمریٹس آئی ڈی',  
 'Start Scanning': 'اسکیننگ شروع کریں',  
 'Card Number': 'کارڈ نمبر',  
 'Card Number not be empty!': 'کارڈ نمبر خالی نہ ہو!',  
 'Please Place the back of your\nTrade License inside the frame':  
 'براہ کرم اپنے تجارتی لائسنس کا \nپچھلا حصہ فریم کے اندر رکھیں',  
 'Scan License': 'اسکین لائسنس',  
 'Please Place the back of your\nEmirates ID inside the frame':  
 'براہ کرم اپنے ایمریٹس آئی ڈی کا\n پچھلا حصہ فریم کے اندر رکھیں',  
 'Scan Card': 'اسکین کارڈ',  
 'Date of Birth': 'پیدائش کی تاریخ',  
 'Date of Birth not be empty!': 'تاریخ پیدائش خالی نہیں ہے!',  
 'Expiry Date': 'میعاد ختم ہونے کی تاریخ',  
 'Expiry Date not be empty!': 'میعاد ختم ہونے کی تاریخ خالی نہیں ہے!',  
 'Gender': 'صنف',  
 'Male': 'مرد',  
 'Female': 'عورت',  
 'Other': 'دیگر',  
 'Gender not be empty!': 'جنس خالی نہیں ہے!',  
 'Please provide your Mobile Phone Number and Email to continue':  
 'جاری رکھنے کے لیے براہ کرم اپنا موبائل فون نمبر اور ای میل فراہم کریں۔',  
 'Phone Number': 'فون نمبر',  
 'Email': 'ای میل',  
 'Please enter the OTP Code\nwe sent you':  
 'براہ کرم وہ OTP کوڈ درج کریں جو ہم نے آپ کو بھیجا ہے۔',  
 'I did not receive the OTP': 'مجھے OTP موصول نہیں ہوا۔',  
 'Resend Code': 'کوڈ پھر سے بھیجو',  
 'Please confirm your\nLicense Information':  
 'براہ کرم اپنے \nلائسنس کی معلومات کی تصدیق کریں۔',  
 'License No': 'لائسنس نمبر',  
 'License No not be empty!': 'لائسنس نہیں خالی نہیں!',  
 'Trade Name': 'تجارتی نام',  
 'Trade Name not be empty!': 'تجارتی نام خالی نہ ہو!',  
 'Scan the back of your\n': 'اپنے پیچھے کو اسکین کریں\n',  
 'Trade License': 'تجارتی لائسنس',  
 'WellDone': 'بہت اچھے',  
 'Congratulations': 'مبارک ہو',  
 'You have successfully registered\nyour ALGORITHMI account':  
 'آپ نے کامیابی سے اپنا ALGORITHMI اکاؤنٹ رجسٹر کر لیا ہے۔',  
 'You have successfully added\nyour':  
 'آپ نے کامیابی سے\nاپنا شامل کر لیا ہے۔',  
 'You\'re upload file and Submitted\nfor review':  
 'آپ نے فائل اپ لوڈ کر دی ہے اور جائزہ کے لیے\nجمع کر دی ہے۔',  
 'Your survey has been Finished and\nSubmitted for review':  
 'آپ کا سروے مکمل ہو چکا ہے اور جائزہ کے لیے جمع کر دیا گیا ہے۔',  
 'Go to Dashboard': 'ڈیش بورڈ پر جائیں',  
 'Dashboard': 'ڈیش بورڈ',  
 'Wallet': 'پرس',  
 'Settings': 'ترتیبات',  
 'Muhammad Usman': 'محمد عثمان',  
 'Verified Account': 'تصدیق شدہ اکاؤنٹ',  
 'Current location: ': 'موجودہ مقام: ',  
 'Al Barai, Dubai': 'البرائی، دبئی',  
 'Account Type: ': 'اکاؤنٹ کی اقسام: ',  
 'Shop Keeper': 'شاپ کیپر',  
 'Member Status: ': 'رکن کی حیثیت: ',  
 'Bronze': 'کانسی',  
 'Questioners: ': 'سائل: ',  
 '3/5 Approved': '3/5 منظور شدہ',  
 'Questionnaire': 'سوالنامہ',  
 '1) Does the location have a POS system.\n2) # of counters.\n3) Sales of tobacco.\n4) Attach image of tobacco shelf.\n5) Ask shop keep top selling brands.\n6) Sales of energy drinks.':  
 '1) کیا مقام پر POS سسٹم ہے؟ \n 2) # کاؤنٹرز۔ \n 3) تمباکو کی فروخت۔ \n 4) تمباکو کے شیلف کی تصویر منسلک کریں۔ \n 5) دکان سے پوچھیں کہ سب سے زیادہ فروخت ہونے والے برانڈز رکھیں۔ \n 6) انرجی ڈرنکس کی فروخت۔',  
 'Accept': 'قبول کریں۔',  
 'Cancel': 'منسوخ کریں۔',  
 'Start New Survey': 'نیا سروے شروع کریں۔',  
 'Last Survey Conducted on 26 Nov 2022':  
 'آخری سروے 26 نومبر 2022 کو کیا گیا۔',  
 'Upload Invoices': 'رسیدیں اپ لوڈ کریں۔',  
 'Last Invoice Uploaded on 26 Nov 2022':  
 'آخری انوائس 26 نومبر 2022 کو اپ لوڈ کی گئی۔',  
 'Upload Stock Report': 'اسٹاک رپورٹ اپ لوڈ کریں۔',  
 'Last Report Uploaded on 26 Nov 2022':  
 'آخری رپورٹ 26 نومبر 2022 کو اپ لوڈ کی گئی۔',  
 'Let’s Start...': 'شروع کرتے ہیں...',  
 'Quick Supermarket, Majan': 'کوئیک سپر مارکیٹ، مجن',  
 'Your slot has been blocked for\n60 min':  
 'آپ کا سلاٹ \n60 منٹ کے لیے بلاک کر دیا گیا ہے۔',  
 'Start Survey Now!': 'ابھی سروے شروع کریں!',  
 'Upload File': 'اپ لوڈ فائل',  
 'Select File': 'فائل کو منتخب کریں۔',  
 'Question': 'سوال',  
 'Does the location have a POS system?': 'کیا مقام پر POS سسٹم ہے؟',  
 'Total Numbers of cash counters?': 'کیش کاؤنٹرز کی کل تعداد؟',  
 'Do they sale tobacco?': 'کیا وہ تمباکو بیچتے ہیں؟',  
 'Attach image of tobacco shelf.':  
 'تمباکو کے شیلف کی تصویر منسلک کریں۔',  
 'Ask shopkeeper top selling brands?':  
 'دکاندار سب سے زیادہ فروخت ہونے والے برانڈز سے پوچھیں؟',  
 'Take Picture': 'تصویر لے',  
 'Yes': 'جی ہاں',  
 'No': 'نہیں',  
 'Select Brand': 'برانڈ منتخب کریں۔',  
 'Nike': 'نائکی',  
 'Hermes': 'ہرمیس',  
 'Louis Vuitton': 'لوئس ووٹن',  
 'Select Option': 'اختیار منتخب کریں۔',  
 'Select Amount': 'رقم منتخب کریں۔',  
 'Upload Image': 'تصویر انٹرنیٹ پر ڈالنا',  
 'Sign Out': 'باہر جائیں',  
 'Language': 'زبان',  
 'Preferences': 'ترجیحات',  
 'Rate the App': 'ایپ کی درجہ بندی کریں۔',  
 'Help': 'مدد',  
 'Support': 'حمایت',  
 'Rewards': 'انعامات',  
 'You have 568 points': 'آپ کے 568 پوائنٹس ہیں۔',  
 'Benefits': 'فوائد',  
 'Notifications': 'اطلاعات',  
 'Accounts': 'اکاؤنٹس',  
 'Personal Information': 'ذاتی معلومات',  
 'Your Account': 'آپ کا کھاتہ',  
 'Profile': 'پروفائل',  
 'Date Of Birth': 'پیدائش کی تاریخ',  
 'United Arab Emirates': 'متحدہ عرب امارات',  
 'Saved Payment Methods': 'محفوظ شدہ ادائیگی کے طریقے',  
 'Add New': 'نیا شامل کریں',  
 'Add': 'شامل کریں۔',  
 'Add Payment Method': 'ادائیگی کا طریقہ شامل کریں۔',  
 'First Abu Dhabi Bank': 'پہلا ابوظہبی بینک',  
 'Dubai Islamic Bank': 'دبئی اسلامک بینک',  
 'Etisalat Wallet': 'اتصالات والے',  
 'Magnati': 'میگناتی',  
 'Bank Account': 'بینک اکاؤنٹ',  
 'E-Wallet': 'ای والیٹ',  
 'Today\'s Notification': 'آج کا نوٹیفکیشن',  
 'Approved Review': 'منظور شدہ جائزہ',  
 'Your Given Review about Shop is approved from Admin.':  
 'شاپ کے بارے میں آپ کا دیا ہوا جائزہ ایڈمن سے منظور شدہ ہے۔',  
 'Send via crypto network': 'کرپٹو نیٹ ورک کے ذریعے بھیجیں۔',  
 'Send via email/ Phone/ Pay ID': 'ای میل/ فون/ پے آئی ڈی کے ذریعے بھیجیں۔',  
 'send to a known crypto address\nvia crypto network':  
 'کرپٹو نیٹ ورک کے ذریعے ایک معروف کرپٹو\n ایڈریس پر بھیجیں۔',  
 'Recipient must be a Algorthmi user':  
 'وصول کنندہ ایک الگورتھمی صارف ہونا چاہیے۔',  
 'Coins': 'سکے',  
 'Cash': 'نقد',  
 'You currently have': 'آپ کے پاس فی الحال ہے۔',  
 'Link Account': 'اکاؤنٹ لنک کریں۔',  
 'Transfer to Bank/E-Wallet': 'بینک / ای والیٹ میں منتقل کریں۔t',  
 'Withdraw': 'واپس لینا',  
 'Transfer': 'منتقلی',  
 'Withdrawal Money': 'رقم نکالنا',  
 'Al Ansari Exchange': 'الانصاری ایکسچینج',  
 },  
 };  
}

# view

## otp\_screen

import 'dart:developer';  
  
import 'package:algorthimi/view/AuthScreens/scan\_id\_freelancer\_screen.dart';  
import 'package:algorthimi/view/Widgets/widget\_utils.dart';  
import 'package:flutter/material.dart';  
import 'package:intl\_phone\_number\_input/intl\_phone\_number\_input.dart';  
import 'dart:async';  
import 'package:firebase\_core/firebase\_core.dart';  
import 'package:firebase\_auth/firebase\_auth.dart';  
import 'package:algorthimi/utils/app\_color.dart';  
import 'package:algorthimi/view/AuthScreens/scan\_id\_camera\_screen.dart';  
import 'package:algorthimi/view/AuthScreens/welldone\_screen.dart';  
import 'package:email\_validator/email\_validator.dart';  
import 'package:flutter/cupertino.dart';  
import 'package:flutter/widgets.dart';  
import 'package:get/get.dart';  
import 'package:pin\_code\_fields/pin\_code\_fields.dart';  
import 'package:responsive\_sizer/responsive\_sizer.dart';  
  
import '../../utils/color\_data.dart';  
  
class OtpScreen extends StatefulWidget {  
 const OtpScreen({super.key});  
  
 @override  
 State<OtpScreen> createState() => \_OtpScreenState();  
}  
  
class \_OtpScreenState extends State<OtpScreen> {  
 final TextEditingController \_phoneNumberController = TextEditingController();  
 final TextEditingController emailController = TextEditingController();  
 final TextEditingController \_otpController = TextEditingController();  
 final FirebaseAuth \_auth = FirebaseAuth.instance;  
 String \_verificationId = '';  
 String phoneNumbers = '';  
  
 Future<void> verifyPhoneNumber(BuildContext context) async {  
 PhoneVerificationCompleted verificationCompleted = (PhoneAuthCredential phoneAuthCredential) async {  
 await \_auth.signInWithCredential(phoneAuthCredential);  
 Get.snackbar(  
 'Processing',  
 'Phone number automatically verified.',  
 duration: const Duration(seconds: 3),  
 backgroundColor: const Color(pink),  
 colorText: Colors.white,  
 );  
 };  
  
 PhoneVerificationFailed verificationFailed =  
 (FirebaseAuthException authException) {  
 Get.snackbar(  
 'Error',  
 'Phone number verification failed.',  
 duration: const Duration(seconds: 3),  
 backgroundColor: const Color(pink),  
 colorText: Colors.white,  
 );  
 };  
  
 PhoneCodeSent codeSent =  
 (String verificationId, [int? forceResendingToken]) async {  
 Get.snackbar(  
 'Success',  
 'Please check your phone for the verification code.',  
 duration: const Duration(seconds: 3),  
 backgroundColor: const Color(pink),  
 colorText: Colors.white,  
 );  
 \_verificationId = verificationId;  
 };  
  
 PhoneCodeAutoRetrievalTimeout codeAutoRetrievalTimeout =  
 (String verificationId) {  
 \_verificationId = verificationId;  
 };  
  
 try {  
 await \_auth.verifyPhoneNumber(  
 phoneNumber: phoneNumbers,  
 timeout: const Duration(seconds: 60),  
 verificationCompleted: verificationCompleted,  
 verificationFailed: verificationFailed,  
 codeSent: codeSent,  
 codeAutoRetrievalTimeout: codeAutoRetrievalTimeout,  
 );  
 } catch (e) {  
 Get.snackbar(  
 'Error',  
 'Failed to Verify Phone Number: $e',  
 duration: const Duration(seconds: 3),  
 backgroundColor: const Color(pink),  
 colorText: Colors.white,  
 );  
 }  
 }  
  
 void signInWithPhoneNumber(BuildContext context) async {  
 try {  
 final AuthCredential credential = PhoneAuthProvider.credential(  
 verificationId: \_verificationId,  
 smsCode: \_otpController.text,  
 );  
 await \_auth.signInWithCredential(credential);  
 Get.snackbar(  
 'Success',  
 'Phone number successfully verified.',  
 duration: const Duration(seconds: 3),  
 backgroundColor: const Color(pink),  
 colorText: Colors.white,  
 );  
 } catch (e) {  
 Get.snackbar(  
 'Error',  
 'Failed to verify phone number: $e',  
 duration: const Duration(seconds: 3),  
 backgroundColor: const Color(pink),  
 colorText: Colors.white,  
 );  
 }  
 }  
  
  
 StreamController<ErrorAnimationType>? errorController;  
 String initialCountry = 'AE';  
 PhoneNumber number = PhoneNumber(isoCode: 'AE');  
  
 var currentStep = 3.obs;  
 StepperType stepperType = StepperType.horizontal;  
  
 back() {  
 isScannedFreelancerId.value = false;  
 isScannedShopOwnerId.value = false;  
 isScannedShopOwnerLicense.value = false;  
 Get.back();  
 // Get.back();  
 }  
  
 @override  
 Widget build(BuildContext context) {  
 return Scaffold(  
 appBar: AppBar(  
 centerTitle: true,  
 title: const Text(  
 'Contact info',  
 style: TextStyle(fontWeight: FontWeight.w600, color: Colors.white),  
 ),  
 backgroundColor: const Color(purple),  
 iconTheme: const IconThemeData(color: Colors.white),  
 ),  
 body: SingleChildScrollView(  
 child: Column(  
 mainAxisAlignment: MainAxisAlignment.start,  
 children: <Widget>[  
 const SizedBox(  
 height: 80,  
 ),  
 getCustomTextW6S20(  
 text: 'Please enter your required information',  
 color: Colors.black),  
 const SizedBox(  
 height: 100,  
 ),  
 Padding(  
 padding: const EdgeInsets.fromLTRB(30, 5, 10, 10),  
 child: InternationalPhoneNumberInput(  
 onInputChanged: (PhoneNumber num) {  
 number = num;  
 phoneNumbers = number.phoneNumber!;  
 log(number.phoneNumber.toString());  
 print(  
 "phone number input ${number} ${\_phoneNumberController}");  
 },  
 onInputValidated: (bool value) {  
 log(value.toString());  
 },  
 selectorConfig: const SelectorConfig(  
 selectorType: PhoneInputSelectorType.DIALOG,  
 ),  
 ignoreBlank: false,  
 autoValidateMode: AutovalidateMode.disabled,  
 selectorTextStyle: const TextStyle(color: Colors.black),  
 initialValue: number,  
 textFieldController: \_phoneNumberController,  
 formatInput: true,  
 keyboardType: const TextInputType.numberWithOptions(  
 signed: true, decimal: true),  
 isEnabled: true,  
 inputDecoration: InputDecoration(  
 enabledBorder: UnderlineInputBorder(  
 borderSide:  
 BorderSide(color: backgroundColor, width: 1.0.px),  
 borderRadius: BorderRadius.circular(10.0.px)),  
 focusedBorder: UnderlineInputBorder(  
 borderSide:  
 BorderSide(color: backgroundColor, width: 1.0.px),  
 borderRadius: BorderRadius.circular(10.0.px)),  
 border: UnderlineInputBorder(  
 borderRadius: BorderRadius.circular(10.0.px)),  
 fillColor: backgroundColor,  
 hintText: ' Enter your Number'.tr,  
 hintStyle: getCustomTextStyleW5S15(  
 color: textBlackColor,  
 ),  
 enabled: true,  
 ),  
 cursorColor: textBlackColor,  
 onSaved: (PhoneNumber number) {  
 log('On Saved: $number');  
 },  
 ),  
 ),  
 Divider(  
 color: subTextColor,  
 thickness: 2,  
 ),  
 Padding(  
 padding: const EdgeInsets.fromLTRB(30, 5, 10, 10),  
 child: Row(  
 children: [  
 Expanded(  
 flex: 1,  
 child: getCustomTextW6S15(  
 text: 'Email'.tr, color: Colors.black),  
 ),  
 Expanded(  
 flex: 2,  
 child: getCustomTextFormField(  
 keyboardType: TextInputType.emailAddress,  
 obscureText: false,  
 hintText: 'Enter Your Email'.tr,  
 controller: emailController,  
 validator: (value) => EmailValidator.validate(value!)  
 ? null  
 : 'Please enter a valid email'.tr,  
 )),  
 ],  
 ),  
 ),  
 const SizedBox(  
 height: 150,  
 ),  
 Padding(  
 padding: const EdgeInsets.all(20.0),  
 child: GestureDetector(  
 onTap: () {  
 print('phone controller value ${phoneNumbers}');  
 verifyPhoneNumber(context);  
 Get.to(() => const ScanIdFreelancerScreen());  
 },  
 child: fillColorButton(  
 color: const Color(pink),  
 text: 'Next'.tr,  
 ),  
 ),  
 ),  
 ],  
 ),  
 ),  
 );  
 }  
}

# AuthScreens

## forgot\_password\_confirmed\_screen

import 'package:algorthimi/view/AuthScreens/login\_screen.dart';  
import 'package:flutter/material.dart';  
import 'package:get/get.dart';  
import 'package:responsive\_sizer/responsive\_sizer.dart';  
  
import '../../utils/color\_data.dart';  
import '../Widgets/widget\_utils.dart';  
  
class ForgotPasswordConfirmedScreen extends StatefulWidget {  
 const ForgotPasswordConfirmedScreen({Key? key}) : super(key: key);  
  
 @override  
 State<ForgotPasswordConfirmedScreen> createState() =>  
 \_ForgotPasswordConfirmedScreenState();  
}  
  
class \_ForgotPasswordConfirmedScreenState  
 extends State<ForgotPasswordConfirmedScreen> {  
 TextEditingController passwordController = TextEditingController();  
 TextEditingController password1Controller = TextEditingController();  
 var isShowPassword = true.obs;  
 var isShowPassword1 = true.obs;  
  
 @override  
 Widget build(BuildContext context) {  
 return Obx(  
 () => Scaffold(  
 resizeToAvoidBottomInset: false,  
 backgroundColor: backgroundColor,  
 body: SizedBox(  
 height: 100.h,  
 width: 100.h,  
 child: Stack(  
 children: [  
 Container(  
 height: 22.h,  
 width: 100.w,  
 alignment: Alignment.topCenter,  
 padding: EdgeInsets.only(top: 6.h, left: 3.h, right: 3.h),  
 decoration: BoxDecoration(  
 color: accentColor,  
 borderRadius:  
 BorderRadius.only(bottomRight: Radius.circular(4.h))),  
 child: Row(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 mainAxisAlignment: MainAxisAlignment.start,  
 children: [  
 GestureDetector(  
 onTap: () {  
 Get.back();  
 },  
 child: Icon(Icons.arrow\_back\_ios\_new\_outlined,  
 color: backgroundColor, size: 2.5.h),  
 ),  
 const Spacer(),  
 getCustomTextW6S22(  
 text: 'Reset Password'.tr,  
 color: textWhiteColor,  
 ),  
 const Spacer()  
 ],  
 ),  
 ),  
 Positioned(  
 top: 16.h,  
 right: 0,  
 left: 0,  
 child: Container(  
 width: 100.h,  
 height: 100.h,  
 padding: EdgeInsets.only(top: 2.h),  
 alignment: Alignment.topCenter,  
 margin: EdgeInsets.symmetric(horizontal: 3.h),  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius: BorderRadius.all(Radius.circular(10.px))),  
 child: Column(  
 children: <Widget>[  
 Padding(  
 padding: EdgeInsets.symmetric(horizontal: 3.h),  
 child: getCustomTextW6S15(  
 text:  
 'Your new password must be different from previous used passwords.'.tr,  
 color: textBlackColor,  
 textAlign: TextAlign.center),  
 ),  
 getVerSpace(10.h),  
 Container(  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius:  
 BorderRadius.all(Radius.circular(15.px)),  
 boxShadow: [  
 BoxShadow(  
 color: borderColor,  
 blurRadius: 10,  
 offset: const Offset(  
 0, 0), // changes position of shadow  
 ),  
 ]),  
 child: getCustomTextFormField(  
 keyboardType: TextInputType.visiblePassword,  
 obscureText: isShowPassword.value,  
 controller: passwordController,  
 hintText: 'Password'.tr,  
 validator: (value) {  
 if (value!.isEmpty) {  
 return "Password not be empty!".tr;  
 } else if (value.length <= 4) {  
 return "Password must be greater than 5 character!"  
 .tr;  
 } else {  
 return null;  
 }  
 },  
 suffixIcon: GestureDetector(  
 onTap: () {  
 isShowPassword.value = !isShowPassword.value;  
 },  
 child: isShowPassword.value  
 ? Icon(  
 Icons.remove\_red\_eye\_sharp,  
 color: textBlackColor,  
 size: 20.px,  
 )  
 : Icon(  
 Icons.password,  
 color: textBlackColor,  
 size: 20.px,  
 )),  
 ),  
 ),  
 getVerSpace(2.h),  
 Container(  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius:  
 BorderRadius.all(Radius.circular(15.px)),  
 boxShadow: [  
 BoxShadow(  
 color: borderColor,  
 blurRadius: 10,  
 offset: const Offset(  
 0, 0), // changes position of shadow  
 ),  
 ]),  
 child: getCustomTextFormField(  
 keyboardType: TextInputType.visiblePassword,  
 obscureText: isShowPassword1.value,  
 controller: password1Controller,  
 hintText: 'Confirm Password'.tr,  
 validator: (value) {  
 if (value!.isEmpty) {  
 return "Password not be empty!".tr;  
 } else if (value.length <= 4) {  
 return "Password must be greater than 5 character!"  
 .tr;  
 } else {  
 return null;  
 }  
 },  
 suffixIcon: GestureDetector(  
 onTap: () {  
 isShowPassword1.value = !isShowPassword1.value;  
 },  
 child: isShowPassword1.value  
 ? Icon(  
 Icons.remove\_red\_eye\_sharp,  
 color: textBlackColor,  
 size: 20.px,  
 )  
 : Icon(  
 Icons.password,  
 color: textBlackColor,  
 size: 20.px,  
 )),  
 ),  
 ),  
 getVerSpace(3.h),  
 GestureDetector(  
 onTap: () {  
 Get.to(() => const LoginScreen());  
 },  
 child: fillColorButton(  
 color: pinkAppColor, text: 'Reset Password'.tr)),  
 ],  
 ),  
 ),  
 ),  
 ],  
 ),  
 ),  
 ),  
 );  
 }  
}

# AuthScreens

## forgot\_password\_screen

import 'package:email\_validator/email\_validator.dart';  
import 'package:flutter/material.dart';  
import 'package:get/get.dart';  
import 'package:responsive\_sizer/responsive\_sizer.dart';  
  
import '../../utils/color\_data.dart';  
import '../Widgets/widget\_utils.dart';  
import 'forgot\_password\_confirmed\_screen.dart';  
  
class ForgotPasswordScreen extends StatefulWidget {  
 const ForgotPasswordScreen({Key? key}) : super(key: key);  
  
 @override  
 State<ForgotPasswordScreen> createState() => \_ForgotPasswordScreenState();  
}  
  
class \_ForgotPasswordScreenState extends State<ForgotPasswordScreen> {  
 TextEditingController emailController = TextEditingController();  
  
 @override  
 Widget build(BuildContext context) {  
 return Scaffold(  
 resizeToAvoidBottomInset: false,  
 backgroundColor: backgroundColor,  
 body: SizedBox(  
 height: 100.h,  
 width: 100.h,  
 child: Stack(  
 children: [  
 Container(  
 height: 22.h,  
 width: 100.w,  
 alignment: Alignment.topCenter,  
 padding: EdgeInsets.only(top: 6.h, left: 3.h, right: 3.h),  
 decoration: BoxDecoration(  
 color: accentColor,  
 borderRadius:  
 BorderRadius.only(bottomRight: Radius.circular(4.h))),  
 child: Row(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 mainAxisAlignment: MainAxisAlignment.start,  
 children: [  
 GestureDetector(  
 onTap: () {  
 Get.back();  
 },  
 child: Icon(Icons.arrow\_back\_ios\_new\_outlined,  
 color: backgroundColor, size: 2.5.h),  
 ),  
 const Spacer(),  
 getCustomTextW6S22(  
 text: 'Forgot Password'.tr,  
 color: textWhiteColor,  
 ),  
 const Spacer()  
 ],  
 ),  
 ),  
 Positioned(  
 top: 16.h,  
 right: 0,  
 left: 0,  
 child: Container(  
 width: 100.h,  
 height: 100.h,  
 padding: EdgeInsets.only(top: 2.h),  
 alignment: Alignment.topCenter,  
 margin: EdgeInsets.symmetric(horizontal: 3.h),  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius: BorderRadius.all(Radius.circular(10.px))),  
 child: Column(  
 children: <Widget>[  
 Padding(  
 padding: EdgeInsets.symmetric(horizontal: 3.h),  
 child: getCustomTextW6S15(  
 text:  
 'Don’t worry! It happens. Please enter the email address associated with your account.'.tr,  
 color: textBlackColor,  
 textAlign: TextAlign.center),  
 ),  
 getVerSpace(10.h),  
 Container(  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius:  
 BorderRadius.all(Radius.circular(15.px)),  
 boxShadow: [  
 BoxShadow(  
 color: borderColor,  
 blurRadius: 10,  
 offset: const Offset(  
 0, 0), // changes position of shadow  
 ),  
 ]),  
 child: getCustomTextFormField(  
 keyboardType: TextInputType.emailAddress,  
 obscureText: false,  
 hintText: 'Enter Your Email'.tr,  
 controller: emailController,  
 validator: (value) => EmailValidator.validate(value!)  
 ? null  
 : "Please enter a valid email".tr,  
 onChanged: (value) {},  
 ),  
 ),  
 getVerSpace(3.h),  
 GestureDetector(  
 onTap: () {  
 Get.to(() => const ForgotPasswordConfirmedScreen());  
 },  
 child:  
 fillColorButton(color: pinkAppColor, text: 'Submit'.tr)),  
 ],  
 ),  
 ),  
 ),  
 ],  
 ),  
 ),  
 );  
 }  
}

# AuthScreens

## login\_screen

import 'package:algorthimi/view/BottomNavBar/bottom\_nav\_bar\_freelancer\_screen.dart';  
import 'package:algorthimi/view/BottomNavBar/bottom\_nav\_bar\_shop\_owner\_screen.dart';  
import 'package:email\_validator/email\_validator.dart';  
import 'package:flutter/material.dart';  
import 'package:get/get.dart';  
import 'package:responsive\_sizer/responsive\_sizer.dart';  
  
import '../../utils/color\_data.dart';  
import '../Widgets/widget\_utils.dart';  
import 'forgot\_password\_screen.dart';  
import 'sign\_up\_screen.dart';  
  
class LoginScreen extends StatefulWidget {  
 const LoginScreen({Key? key}) : super(key: key);  
  
 @override  
 State<LoginScreen> createState() => \_LoginScreenState();  
}  
  
class \_LoginScreenState extends State<LoginScreen> {  
 TextEditingController emailController = TextEditingController();  
 TextEditingController passwordController = TextEditingController();  
 var isShowPassword = true.obs;  
 var isCheck = false.obs;  
 var isSigningIn = false.obs;  
  
 @override  
 Widget build(BuildContext context) {  
 return Obx(  
 () => Scaffold(  
 resizeToAvoidBottomInset: false,  
 backgroundColor: backgroundColor,  
 body: Container(  
 width: 100.h,  
 height: 100.h - 10.h,  
 padding:  
 EdgeInsets.only(top: 16.h, right: 3.h, left: 3.h, bottom: 10.h),  
 decoration: BoxDecoration(  
 color: accentColor,  
 borderRadius:  
 BorderRadius.only(bottomRight: Radius.circular(20.h))),  
 child: Column(  
 mainAxisAlignment: MainAxisAlignment.spaceBetween,  
 crossAxisAlignment: CrossAxisAlignment.center,  
 children: [  
 Column(  
 children: [  
 Align(  
 alignment: Alignment.center,  
 child: getSvgImage('app\_logo.svg', height: 12.h)),  
 getVerSpace(1.h),  
 getCustomTextW6S22(text: 'Login'.tr, color: textWhiteColor),  
 ],  
 ),  
 Column(  
 children: [  
 getCustomTextFormField(  
 keyboardType: TextInputType.emailAddress,  
 obscureText: false,  
 hintText: 'Enter Your Email'.tr,  
 controller: emailController,  
 validator: (value) => EmailValidator.validate(value!)  
 ? null  
 : "Please enter a valid email".tr,  
 ),  
  
 getVerSpace(2.5.h),  
  
 getCustomTextFormField(  
 keyboardType: TextInputType.visiblePassword,  
 obscureText: isShowPassword.value,  
 controller: passwordController,  
 hintText: 'Enter Your Password'.tr,  
 validator: (value) {  
 if (value!.isEmpty) {  
 return "Password not be empty!".tr;  
 } else if (value.length <= 4) {  
 return "Password must be greater than 5 character!".tr;  
 } else {  
 return null;  
 }  
 },  
 suffixIcon: GestureDetector(  
 onTap: () {  
 isShowPassword.value = !isShowPassword.value;  
 },  
 child: isShowPassword.value  
 ? Icon(  
 Icons.remove\_red\_eye\_sharp,  
 color: textBlackColor,  
 size: 20.px,  
 )  
 : Icon(  
 Icons.password,  
 color: textBlackColor,  
 size: 20.px,  
 )),  
 ),  
  
 getVerSpace(2.5.h),  
  
 GestureDetector(  
 onTap: () {  
 Get.to(() => const ForgotPasswordScreen());  
 },  
 child: getCustomTextW6S15(  
 text: 'Forgot Password?'.tr, color: textWhiteColor),  
 ),  
  
 getVerSpace(3.h),  
  
 GestureDetector(  
 onTap: () {  
 // Get.to(() => const BottomNavBarFreelancerScreen());  
 Get.to(() => const BottomNavBarShopOwnerScreen());  
 },  
 child:  
 fillColorButton(color: pinkAppColor, text: 'Login'.tr)),  
 ],  
 ),  
 GestureDetector(  
 onTap: () {  
 Get.to(() => const SignUpScreen());  
 },  
 child: RichText(  
 textAlign: TextAlign.center,  
 text: TextSpan(  
 text: 'Don’t have an account ? '.tr,  
 style: getCustomTextStyleW5S15(  
 color: textWhiteColor,  
 ),  
 children: [  
 TextSpan(  
 text: 'Sign Up'.tr,  
 style: getCustomTextStyleW7S15(  
 color: skyBlueTextColor,  
 ),  
 )  
 ]),  
 ),  
 )  
 ],  
 ),  
 ),  
 ),  
 );  
 }  
}

# AuthScreens

## scan\_id\_camera\_screen

import 'dart:async';  
  
import 'package:algorthimi/view/Widgets/widget\_utils.dart';  
import 'package:flutter/material.dart';  
import 'package:get/get.dart';  
import 'package:responsive\_sizer/responsive\_sizer.dart';  
import 'dart:io';  
import 'package:camera/camera.dart';  
import 'package:google\_mlkit\_text\_recognition/google\_mlkit\_text\_recognition.dart';  
import 'package:permission\_handler/permission\_handler.dart';  
  
import '../../utils/color\_data.dart';  
  
String? scanText;  
var isScannedFreelancerId = false.obs;  
var isScannedShopOwnerId = false.obs;  
var isScannedShopOwnerLicense = false.obs;  
  
class ScanIdCameraScreen extends StatefulWidget {  
 const ScanIdCameraScreen({Key? key}) : super(key: key);  
  
 @override  
 State<ScanIdCameraScreen> createState() => \_ScanIdCameraScreenState();  
}  
  
class \_ScanIdCameraScreenState extends State<ScanIdCameraScreen> {  
 @override  
 void initState() {  
 // TODO: implement initState  
 super.initState();  
 Timer(const Duration(milliseconds: 500), () {  
 Get.to(() => const ScannerScreen());  
 });  
 }  
  
 @override  
 Widget build(BuildContext context) {  
 return Scaffold(  
 resizeToAvoidBottomInset: false,  
 backgroundColor: Colors.grey,  
 body: SizedBox(  
 height: 100.h,  
 width: 100.w,  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 mainAxisAlignment: MainAxisAlignment.center,  
 children: [  
 getCustomTextW6S16(  
 text:  
 'Please Place the back of your\nEmirates ID inside the frame'.tr,  
 color: textWhiteColor,  
 textAlign: TextAlign.center),  
 getVerSpace(10.h),  
 getSvgImage('scan\_id\_white.svg', height: 19.h)  
 ],  
 ),  
 ),  
 );  
 }  
}  
  
class ScannerScreen extends StatefulWidget {  
 const ScannerScreen({super.key});  
  
 @override  
 State<ScannerScreen> createState() => \_ScannerScreenState();  
}  
  
// Add the WidgetsBindingObserver mixin  
class \_ScannerScreenState extends State<ScannerScreen>  
 with WidgetsBindingObserver {  
 bool \_isPermissionGranted = false;  
 final textRecognizer = TextRecognizer();  
  
 late final Future<void> \_future;  
  
 // Add this controller to be able to control de camera  
 CameraController? \_cameraController;  
  
 back() {  
 Get.back();  
 Get.back();  
 }  
  
 @override  
 void initState() {  
 super.initState();  
 WidgetsBinding.instance.addObserver(this);  
  
 \_future = \_requestCameraPermission();  
 }  
  
 // We should stop the camera once this widget is disposed  
 @override  
 void dispose() {  
 WidgetsBinding.instance.removeObserver(this);  
 \_stopCamera();  
 textRecognizer.close();  
 super.dispose();  
 }  
  
 // Starts and stops the camera according to the lifecycle of the app  
 @override  
 void didChangeAppLifecycleState(AppLifecycleState state) {  
 if (\_cameraController == null || !\_cameraController!.value.isInitialized) {  
 return;  
 }  
  
 if (state == AppLifecycleState.inactive) {  
 \_stopCamera();  
 } else if (state == AppLifecycleState.resumed &&  
 \_cameraController != null &&  
 \_cameraController!.value.isInitialized) {  
 \_startCamera();  
 }  
 }  
  
 @override  
 Widget build(BuildContext context) {  
 return WillPopScope(  
 onWillPop: () {  
 return back();  
 },  
 child: FutureBuilder(  
 future: \_future,  
 builder: (context, snapshot) {  
 return Stack(  
 children: [  
 // Show the camera feed behind everything  
 if (\_isPermissionGranted)  
 FutureBuilder<List<CameraDescription>>(  
 future: availableCameras(),  
 builder: (context, snapshot) {  
 if (snapshot.hasData) {  
 \_initCameraController(snapshot.data!);  
  
 return Center(child: CameraPreview(\_cameraController!));  
 } else {  
 return const LinearProgressIndicator();  
 }  
 },  
 ),  
 Scaffold(  
 // Set the background to transparent so you can see the camera preview  
 backgroundColor:  
 \_isPermissionGranted ? Colors.transparent : null,  
 body: \_isPermissionGranted  
 ? Column(  
 children: [  
 Expanded(  
 child: Container(),  
 ),  
 Container(  
 padding: const EdgeInsets.only(bottom: 30.0),  
 child: Center(  
 child: OutlinedButton(  
 style: ButtonStyle(  
 backgroundColor: MaterialStatePropertyAll(  
 backgroundColor),  
 overlayColor: MaterialStatePropertyAll(  
 pinkAppColor.withOpacity(0.1)),  
 side: MaterialStateProperty.all(BorderSide(  
 color: pinkAppColor,  
 width: 1,  
 style: BorderStyle.solid)),  
 shape: MaterialStatePropertyAll(  
 RoundedRectangleBorder(  
 borderRadius: BorderRadius.all(  
 Radius.circular(10.px)))),  
 padding: MaterialStatePropertyAll(  
 EdgeInsets.symmetric(horizontal: 3.h))),  
 onPressed: \_scanImage,  
 child: getCustomTextW6S16(text: 'Scan Card'.tr, color: pinkAppColor))),  
 ),  
 ],  
 )  
 : Center(  
 child: Container(  
 padding:  
 const EdgeInsets.only(left: 24.0, right: 24.0),  
 child: const Text(  
 'Camera permission denied',  
 textAlign: TextAlign.center,  
 ),  
 ),  
 ),  
 ),  
 ],  
 );  
 },  
 ),  
 );  
 }  
  
 Future<void> \_requestCameraPermission() async {  
 final status = await Permission.camera.request();  
 \_isPermissionGranted = status == PermissionStatus.granted;  
 }  
  
 void \_startCamera() {  
 if (\_cameraController != null) {  
 \_cameraSelected(\_cameraController!.description);  
 }  
 }  
  
 void \_stopCamera() {  
 if (\_cameraController != null) {  
 \_cameraController?.dispose();  
 }  
 }  
  
 void \_initCameraController(List<CameraDescription> cameras) {  
 if (\_cameraController != null) {  
 return;  
 }  
  
 // Select the first rear camera.  
 CameraDescription? camera;  
 for (var i = 0; i < cameras.length; i++) {  
 final CameraDescription current = cameras[i];  
 if (current.lensDirection == CameraLensDirection.back) {  
 camera = current;  
 break;  
 }  
 }  
  
 if (camera != null) {  
 \_cameraSelected(camera);  
 }  
 }  
  
 Future<void> \_cameraSelected(CameraDescription camera) async {  
 \_cameraController = CameraController(  
 camera,  
 ResolutionPreset.max,  
 enableAudio: false,  
 );  
  
 await \_cameraController!.initialize();  
  
 if (!mounted) {  
 return;  
 }  
 setState(() {});  
 }  
  
 Future<void> \_scanImage() async {  
  
 isScannedFreelancerId.value = true;  
 isScannedShopOwnerId.value = true;  
  
  
 if (\_cameraController == null) return;  
 try {  
 final pictureFile = await \_cameraController!.takePicture();  
  
 final file = File(pictureFile.path);  
  
 final inputImage = InputImage.fromFile(file);  
 final recognizedText = await textRecognizer.processImage(inputImage);  
  
 scanText = recognizedText.text;  
 Get.back();  
 Get.back();  
 } catch (e) {  
 ScaffoldMessenger.of(context).showSnackBar(  
 const SnackBar(  
 content: Text('An error occurred when scanning text'),  
 ),  
 );  
 }  
 }  
}

# AuthScreens

## scan\_id\_camera\_shop\_owner\_screen

import 'dart:async';  
  
import 'package:algorthimi/view/AuthScreens/scan\_id\_camera\_screen.dart';  
import 'package:algorthimi/view/Widgets/widget\_utils.dart';  
import 'package:flutter/material.dart';  
import 'package:get/get.dart';  
import 'package:responsive\_sizer/responsive\_sizer.dart';  
import 'dart:io';  
import 'package:camera/camera.dart';  
import 'package:google\_mlkit\_text\_recognition/google\_mlkit\_text\_recognition.dart';  
import 'package:permission\_handler/permission\_handler.dart';  
  
import '../../utils/color\_data.dart';  
  
String? scanText;  
  
class ScanIdCameraShopOwnerScreen extends StatefulWidget {  
 const ScanIdCameraShopOwnerScreen({Key? key}) : super(key: key);  
  
 @override  
 State<ScanIdCameraShopOwnerScreen> createState() =>  
 \_ScanIdCameraShopOwnerScreenState();  
}  
  
class \_ScanIdCameraShopOwnerScreenState  
 extends State<ScanIdCameraShopOwnerScreen> {  
 @override  
 void initState() {  
 // TODO: implement initState  
 super.initState();  
 Timer(const Duration(milliseconds: 500), () {  
 Get.to(() => const ScannerScreen());  
 });  
 }  
  
 @override  
 Widget build(BuildContext context) {  
 return Scaffold(  
 resizeToAvoidBottomInset: false,  
 backgroundColor: Colors.grey,  
 body: SizedBox(  
 height: 100.h,  
 width: 100.w,  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 mainAxisAlignment: MainAxisAlignment.center,  
 children: [  
 getCustomTextW6S16(  
 text:  
 'Please Place the back of your\nTrade License inside the frame'.tr,  
 color: textWhiteColor,  
 textAlign: TextAlign.center),  
 getVerSpace(10.h),  
 getSvgImage('scan\_id\_white\_shop\_owner.svg', height: 40.h)  
 ],  
 ),  
 ),  
 );  
 }  
}  
  
class ScannerScreen extends StatefulWidget {  
 const ScannerScreen({super.key});  
  
 @override  
 State<ScannerScreen> createState() => \_ScannerScreenState();  
}  
  
// Add the WidgetsBindingObserver mixin  
class \_ScannerScreenState extends State<ScannerScreen>  
 with WidgetsBindingObserver {  
 bool \_isPermissionGranted = false;  
 final textRecognizer = TextRecognizer();  
  
 late final Future<void> \_future;  
  
 // Add this controller to be able to control de camera  
 CameraController? \_cameraController;  
  
 back() {  
 Get.back();  
 Get.back();  
 }  
  
 @override  
 void initState() {  
 super.initState();  
 WidgetsBinding.instance.addObserver(this);  
  
 \_future = \_requestCameraPermission();  
 }  
  
 // We should stop the camera once this widget is disposed  
 @override  
 void dispose() {  
 WidgetsBinding.instance.removeObserver(this);  
 \_stopCamera();  
 textRecognizer.close();  
 super.dispose();  
 }  
  
 // Starts and stops the camera according to the lifecycle of the app  
 @override  
 void didChangeAppLifecycleState(AppLifecycleState state) {  
 if (\_cameraController == null || !\_cameraController!.value.isInitialized) {  
 return;  
 }  
  
 if (state == AppLifecycleState.inactive) {  
 \_stopCamera();  
 } else if (state == AppLifecycleState.resumed &&  
 \_cameraController != null &&  
 \_cameraController!.value.isInitialized) {  
 \_startCamera();  
 }  
 }  
  
 @override  
 Widget build(BuildContext context) {  
 return WillPopScope(  
 onWillPop: () {  
 return back();  
 },  
 child: FutureBuilder(  
 future: \_future,  
 builder: (context, snapshot) {  
 return Stack(  
 children: [  
 // Show the camera feed behind everything  
 if (\_isPermissionGranted)  
 FutureBuilder<List<CameraDescription>>(  
 future: availableCameras(),  
 builder: (context, snapshot) {  
 if (snapshot.hasData) {  
 \_initCameraController(snapshot.data!);  
  
 return Center(child: CameraPreview(\_cameraController!));  
 } else {  
 return const LinearProgressIndicator();  
 }  
 },  
 ),  
 Scaffold(  
 // Set the background to transparent so you can see the camera preview  
 backgroundColor:  
 \_isPermissionGranted ? Colors.transparent : null,  
 body: \_isPermissionGranted  
 ? Column(  
 children: [  
 Expanded(  
 child: Container(),  
 ),  
 Container(  
 padding: const EdgeInsets.only(bottom: 30.0),  
 child: Center(  
 child: OutlinedButton(  
 style: ButtonStyle(  
 backgroundColor: MaterialStatePropertyAll(  
 backgroundColor),  
 overlayColor: MaterialStatePropertyAll(  
 pinkAppColor.withOpacity(0.1)),  
 side: MaterialStateProperty.all(BorderSide(  
 color: pinkAppColor,  
 width: 1,  
 style: BorderStyle.solid)),  
 shape: MaterialStatePropertyAll(  
 RoundedRectangleBorder(  
 borderRadius: BorderRadius.all(  
 Radius.circular(10.px)))),  
 padding: MaterialStatePropertyAll(  
 EdgeInsets.symmetric(horizontal: 3.h))),  
 onPressed: \_scanImage,  
 child: getCustomTextW6S16(text: 'Scan License'.tr, color: pinkAppColor))),  
 ),  
 ],  
 )  
 : Center(  
 child: Container(  
 padding:  
 const EdgeInsets.only(left: 24.0, right: 24.0),  
 child: const Text(  
 'Camera permission denied',  
 textAlign: TextAlign.center,  
 ),  
 ),  
 ),  
 ),  
 ],  
 );  
 },  
 ),  
 );  
 }  
  
 Future<void> \_requestCameraPermission() async {  
 final status = await Permission.camera.request();  
 \_isPermissionGranted = status == PermissionStatus.granted;  
 }  
  
 void \_startCamera() {  
 if (\_cameraController != null) {  
 \_cameraSelected(\_cameraController!.description);  
 }  
 }  
  
 void \_stopCamera() {  
 if (\_cameraController != null) {  
 \_cameraController?.dispose();  
 }  
 }  
  
 void \_initCameraController(List<CameraDescription> cameras) {  
 if (\_cameraController != null) {  
 return;  
 }  
  
 // Select the first rear camera.  
 CameraDescription? camera;  
 for (var i = 0; i < cameras.length; i++) {  
 final CameraDescription current = cameras[i];  
 if (current.lensDirection == CameraLensDirection.back) {  
 camera = current;  
 break;  
 }  
 }  
  
 if (camera != null) {  
 \_cameraSelected(camera);  
 }  
 }  
  
 Future<void> \_cameraSelected(CameraDescription camera) async {  
 \_cameraController = CameraController(  
 camera,  
 ResolutionPreset.max,  
 enableAudio: false,  
 );  
  
 await \_cameraController!.initialize();  
  
 if (!mounted) {  
 return;  
 }  
 setState(() {});  
 }  
  
 Future<void> \_scanImage() async {  
 isScannedFreelancerId.value = true;  
 isScannedShopOwnerId.value = true;  
 isScannedShopOwnerLicense.value = true;  
 if (\_cameraController == null) return;  
 try {  
 final pictureFile = await \_cameraController!.takePicture();  
  
 final file = File(pictureFile.path);  
  
 final inputImage = InputImage.fromFile(file);  
 final recognizedText = await textRecognizer.processImage(inputImage);  
  
 scanText = recognizedText.text;  
 Get.back();  
 Get.back();  
 } catch (e) {  
 ScaffoldMessenger.of(context).showSnackBar(  
 const SnackBar(  
 content: Text('An error occurred when scanning text'),  
 ),  
 );  
 }  
 }  
}

# AuthScreens

## scan\_id\_freelancer\_screen

import 'dart:async';  
import 'dart:developer';  
  
import 'package:algorthimi/utils/app\_color.dart';  
import 'package:algorthimi/view/AuthScreens/scan\_id\_camera\_screen.dart';  
import 'package:algorthimi/view/AuthScreens/welldone\_screen.dart';  
import 'package:email\_validator/email\_validator.dart';  
import 'package:flutter/cupertino.dart';  
import 'package:flutter/material.dart';  
import 'package:flutter/widgets.dart';  
import 'package:get/get.dart';  
import 'package:intl\_phone\_number\_input/intl\_phone\_number\_input.dart';  
import 'package:pin\_code\_fields/pin\_code\_fields.dart';  
import 'package:responsive\_sizer/responsive\_sizer.dart';  
  
import '../../utils/color\_data.dart';  
import '../Widgets/widget\_utils.dart';  
  
class ScanIdFreelancerScreen extends StatefulWidget {  
 const ScanIdFreelancerScreen({Key? key}) : super(key: key);  
  
 @override  
 State<ScanIdFreelancerScreen> createState() => \_ScanIdFreelancerScreenState();  
}  
  
class \_ScanIdFreelancerScreenState extends State<ScanIdFreelancerScreen> {  
 TextEditingController idNumberController = TextEditingController();  
 TextEditingController cardNumberController = TextEditingController();  
 TextEditingController fullNameController = TextEditingController();  
 TextEditingController dateOfBrithController = TextEditingController();  
 TextEditingController nationalityController = TextEditingController();  
 TextEditingController genderController = TextEditingController();  
 TextEditingController expiryDateController = TextEditingController();  
 TextEditingController phoneNumberController = TextEditingController();  
 TextEditingController emailController = TextEditingController();  
 TextEditingController otpController = TextEditingController();  
 StreamController<ErrorAnimationType>? errorController;  
 String initialCountry = 'AE';  
 PhoneNumber number = PhoneNumber(isoCode: 'AE');  
  
 var currentStep = 3.obs;  
 StepperType stepperType = StepperType.horizontal;  
  
 back() {  
 isScannedFreelancerId.value = false;  
 isScannedShopOwnerId.value = false;  
 isScannedShopOwnerLicense.value = false;  
 Get.back();  
 // Get.back();  
 }  
  
 @override  
 Widget build(BuildContext context) {  
 return Obx(  
 () => WillPopScope(  
 onWillPop: () {  
 return back();  
 },  
 child: Scaffold(  
 resizeToAvoidBottomInset: false,  
 backgroundColor: backgroundColor,  
 body: SingleChildScrollView(  
 child: SizedBox(  
 height: 100.h,  
 width: 100.h,  
 child: Stack(  
 children: [  
 SingleChildScrollView(  
 child: Container(  
 height: 22.h,  
 width: 100.w,  
 alignment: Alignment.topCenter,  
 padding: EdgeInsets.only(top: 6.h, left: 3.h, right: 3.h),  
 decoration: BoxDecoration(  
 color: accentColor,  
 borderRadius: BorderRadius.only(  
 bottomRight: Radius.circular(4.h))),  
 child: Row(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 mainAxisAlignment: MainAxisAlignment.start,  
 children: [  
 GestureDetector(  
 onTap: () {  
 isScannedFreelancerId.value = false;  
 isScannedShopOwnerId.value = false;  
 isScannedShopOwnerLicense.value = false;  
 Get.back();  
 // Get.back();  
 },  
 child: Icon(Icons.arrow\_back\_ios\_new\_outlined,  
 color: backgroundColor, size: 2.5.h),  
 ),  
 const Spacer(),  
 SizedBox(  
 width: 30.h,  
 child: getCustomTextW6S22(  
 text: currentStep.value == 0 &&  
 !isScannedFreelancerId.value  
 ? 'Scan Your ID'.tr  
 : currentStep.value == 0 ||  
 currentStep.value == 1  
 ? 'Personal Details\nConfirmation'.tr  
 : currentStep.value == 2  
 ? 'Mobile & Email\nConfirmation'.tr  
 : 'Enter Verification Code'.tr,  
 color: textWhiteColor,  
 ),  
 ),  
 const Spacer()  
 ],  
 ),  
 ),  
 ),  
 Positioned(  
 top: 16.h,  
 right: 0,  
 left: 0,  
 child: Container(  
 width: 100.h,  
 height: 100.h,  
 padding: EdgeInsets.only(top: 2.h),  
 alignment: Alignment.topCenter,  
 margin: EdgeInsets.symmetric(horizontal: 3.h),  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius:  
 BorderRadius.all(Radius.circular(10.px))),  
 child: Theme(  
 data: ThemeData(  
 canvasColor: backgroundColor,  
 colorScheme: Theme.of(context)  
 .colorScheme  
 .copyWith(primary: accentColor),  
 ),  
 child: Stepper(  
 type: stepperType,  
 elevation: 0,  
 physics: const NeverScrollableScrollPhysics(),  
 currentStep: currentStep.value,  
 onStepTapped: (step) => tapped(step),  
 onStepContinue: continued,  
 onStepCancel: cancel,  
 controlsBuilder:  
 (BuildContext context, ControlsDetails details) {  
 return Obx(  
 () => isScannedFreelancerId.value  
 ? currentStep.value == 3  
 ? Container()  
 : Row(children: [  
 Expanded(  
 child: ElevatedButton(  
 style: ButtonStyle(  
 backgroundColor:  
 MaterialStatePropertyAll(  
 pinkAppColor),  
 overlayColor:  
 MaterialStatePropertyAll(  
 pinkAppColor  
 .withOpacity(0.1)),  
 side: MaterialStateProperty.all(  
 BorderSide(  
 color: pinkAppColor,  
 width: 1.5,  
 style:  
 BorderStyle.solid)),  
 shape: MaterialStatePropertyAll(  
 RoundedRectangleBorder(  
 borderRadius:  
 BorderRadius.all(  
 Radius.circular(10.px)))),  
 padding: MaterialStatePropertyAll(EdgeInsets.symmetric(vertical: 1.6.h))),  
 onPressed: details.onStepContinue,  
 child: getCustomTextW6S16(  
 maxLines: 1,  
 text: currentStep.value == 1  
 ? 'Confirm'.tr  
 : currentStep.value == 2  
 ? 'Verify'.tr  
 : 'Next'.tr,  
 color: textWhiteColor,  
 ),  
 )),  
 if (currentStep.value != 0)  
 Expanded(  
 child: Row(  
 children: [  
 getHorSpace(2.h),  
 Expanded(  
 child: OutlinedButton(  
 style: ButtonStyle(  
 overlayColor:  
 MaterialStatePropertyAll(  
 pinkAppColor  
 .withOpacity(  
 0.1)),  
 side: MaterialStateProperty  
 .all(BorderSide(  
 color:  
 pinkAppColor,  
 width: 1.5,  
 style: BorderStyle  
 .solid)),  
 shape: MaterialStatePropertyAll(  
 RoundedRectangleBorder(  
 borderRadius:  
 BorderRadius.all(Radius.circular(10.px)))),  
 padding: MaterialStatePropertyAll(EdgeInsets.symmetric(vertical: 1.6.h))),  
 onPressed: details.onStepCancel,  
 child: getCustomTextW6S16(text: 'Back'.tr, color: pinkAppColor)),  
 ),  
 ],  
 ),  
 )  
 ])  
 : Container(),  
 );  
 },  
 steps: <Step>[  
 Step(  
 title: const Text(''),  
 content: isScannedFreelancerId.value  
 ? Column(  
 children: [  
 getCustomTextW6S15(  
 text:  
 'Please confirm your\nPersonal Information'  
 .tr,  
 color: textBlackColor,  
 textAlign: TextAlign.center),  
 getVerSpace(2.h),  
 Row(  
 children: [  
 Expanded(  
 child: getCustomTextW6S15(  
 text: 'ID Number'.tr,  
 color: subTextColor),  
 ),  
 Expanded(  
 flex: 2,  
 child: getCustomTextFormField(  
 keyboardType:  
 const TextInputType  
 .numberWithOptions(  
 signed: true,  
 decimal: true),  
 obscureText: false,  
 hintText: 'ID Number'.tr,  
 controller:  
 idNumberController,  
 validator: (value) {  
 if (value!.isEmpty) {  
 return 'ID Number not be empty!'  
 .tr;  
 } else {  
 return null;  
 }  
 },  
 ))  
 ],  
 ),  
 Divider(color: subTextColor),  
 Row(  
 children: [  
 Expanded(  
 child: getCustomTextW6S15(  
 text: 'Full Name'.tr,  
 color: subTextColor),  
 ),  
 Expanded(  
 flex: 2,  
 child: getCustomTextFormField(  
 keyboardType:  
 TextInputType.text,  
 obscureText: false,  
 hintText: 'Full Name'.tr,  
 controller:  
 fullNameController,  
 validator: (value) {  
 if (value!.isEmpty) {  
 return 'Full Name not be empty!'  
 .tr;  
 } else {  
 return null;  
 }  
 },  
 ))  
 ],  
 ),  
 Divider(color: subTextColor),  
 Row(  
 children: [  
 Expanded(  
 child: getCustomTextW6S15(  
 text: 'Nationality'.tr,  
 color: subTextColor),  
 ),  
 Expanded(  
 flex: 2,  
 child: getCustomTextFormField(  
 keyboardType:  
 TextInputType.text,  
 obscureText: false,  
 hintText: 'Nationality'.tr,  
 controller:  
 nationalityController,  
 validator: (value) {  
 if (value!.isEmpty) {  
 return 'Nationality not be empty!'  
 .tr;  
 } else {  
 return null;  
 }  
 },  
 ))  
 ],  
 ),  
 getVerSpace(2.h),  
 ],  
 )  
 : Column(  
 mainAxisAlignment:  
 MainAxisAlignment.start,  
 crossAxisAlignment:  
 CrossAxisAlignment.center,  
 children: <Widget>[  
 RichText(  
 textAlign: TextAlign.center,  
 text: TextSpan(  
 text: 'Scan the back of your '.tr,  
 style: getCustomTextStyleW5S15(  
 color: textBlackColor,  
 ),  
 children: [  
 TextSpan(  
 text: 'Emirates ID'.tr,  
 style:  
 getCustomTextStyleW7S15(  
 color: textBlackColor,  
 ),  
 ),  
 ]),  
 ),  
 getVerSpace(10.h),  
 getSvgImage('scan\_id.svg',  
 height: 23.h),  
 getVerSpace(10.h),  
 GestureDetector(  
 onTap: () {  
 Get.to(() =>  
 const ScanIdCameraScreen());  
 },  
 child: fillColorButton(  
 color: pinkAppColor,  
 text: 'Start Scanning'.tr,  
 ),  
 ),  
 getVerSpace(2.h),  
 ],  
 ),  
 isActive: currentStep.value >= 0,  
 state: currentStep.value == 0  
 ? StepState.editing  
 : StepState.complete,  
 ),  
  
 Step(  
 title: const Text(''),  
 content: Column(  
 children: <Widget>[  
 getCustomTextW6S15(  
 text:  
 'Please confirm your\nPersonal Information'  
 .tr,  
 color: textBlackColor,  
 textAlign: TextAlign.center),  
 getVerSpace(2.h),  
 Row(  
 children: [  
 Expanded(  
 child: getCustomTextW6S15(  
 text: 'Card Number'.tr,  
 color: subTextColor),  
 ),  
 Expanded(  
 flex: 2,  
 child: getCustomTextFormField(  
 keyboardType: const TextInputType  
 .numberWithOptions(  
 signed: true, decimal: true),  
 obscureText: false,  
 hintText: 'Card Number'.tr,  
 controller: cardNumberController,  
 validator: (value) {  
 if (value!.isEmpty) {  
 return 'Card Number not be empty!'  
 .tr;  
 } else {  
 return null;  
 }  
 },  
 ))  
 ],  
 ),  
 Divider(color: subTextColor),  
 Row(  
 children: [  
 Expanded(  
 child: getCustomTextW6S15(  
 text: 'Date of Birth'.tr,  
 color: subTextColor),  
 ),  
 Expanded(  
 flex: 2,  
 child: getCustomTextFormField(  
 obscureText: false,  
 readOnly: true,  
 onTap: () async {  
 await showCupertinoModalPopup<  
 void>(  
 context: context,  
 builder: (\_) {  
 return Container(  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius:  
 BorderRadius.only(  
 topLeft:  
 Radius.circular(  
 12.px),  
 topRight:  
 Radius.circular(  
 12.px),  
 ),  
 ),  
 height: 25.h,  
 child: CupertinoDatePicker(  
 mode:  
 CupertinoDatePickerMode  
 .date,  
 onDateTimeChanged:  
 (value) {  
 dateOfBrithController  
 .text =  
 value  
 .toString()  
 .substring(  
 0, 10);  
 },  
 ),  
 );  
 },  
 );  
 },  
 hintText: 'Date of Birth'.tr,  
 controller: dateOfBrithController,  
 validator: (value) {  
 if (value!.isEmpty) {  
 return 'Date of Birth not be empty!'  
 .tr;  
 } else {  
 return null;  
 }  
 },  
 ))  
 ],  
 ),  
 Divider(color: subTextColor),  
 Row(  
 children: [  
 Expanded(  
 child: getCustomTextW6S15(  
 text: 'Expiry Date'.tr,  
 color: subTextColor),  
 ),  
 Expanded(  
 flex: 2,  
 child: getCustomTextFormField(  
 obscureText: false,  
 readOnly: true,  
 onTap: () async {  
 await showCupertinoModalPopup<  
 void>(  
 context: context,  
 builder: (\_) {  
 return Container(  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius:  
 BorderRadius.only(  
 topLeft:  
 Radius.circular(  
 12.px),  
 topRight:  
 Radius.circular(  
 12.px),  
 ),  
 ),  
 height: 25.h,  
 child: CupertinoDatePicker(  
 mode:  
 CupertinoDatePickerMode  
 .date,  
 onDateTimeChanged:  
 (value) {  
 expiryDateController  
 .text =  
 value  
 .toString()  
 .substring(  
 0, 10);  
 },  
 ),  
 );  
 },  
 );  
 },  
 hintText: 'Expiry Date'.tr,  
 controller: expiryDateController,  
 validator: (value) {  
 if (value!.isEmpty) {  
 return 'Expiry Date not be empty!'  
 .tr;  
 } else {  
 return null;  
 }  
 },  
 ))  
 ],  
 ),  
 Divider(color: subTextColor),  
 Row(  
 children: [  
 Expanded(  
 child: getCustomTextW6S15(  
 text: 'Gender'.tr,  
 color: subTextColor),  
 ),  
 Expanded(  
 flex: 2,  
 child: getCustomTextFormField(  
 obscureText: false,  
 readOnly: true,  
 onTap: () {  
 showModalBottomSheet<void>(  
 context: context,  
 elevation: 5,  
 builder:  
 (BuildContext context) {  
 return Container(  
 width: 100.w,  
 height: 25.h,  
 padding:  
 EdgeInsets.symmetric(  
 vertical: 2.h),  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius:  
 BorderRadius.only(  
 topLeft: Radius  
 .circular(  
 15.px),  
 topRight: Radius  
 .circular(15  
 .px))),  
 child: Column(  
 crossAxisAlignment:  
 CrossAxisAlignment  
 .start,  
 mainAxisAlignment:  
 MainAxisAlignment  
 .spaceEvenly,  
 children: [  
 GestureDetector(  
 onTap: () {  
 genderController  
 .text =  
 'Male'  
 .toString()  
 .tr;  
 Get.back();  
 },  
 child: Container(  
 width: 100.w,  
 alignment: Alignment  
 .center,  
 margin: EdgeInsets  
 .symmetric(  
 horizontal:  
 3.h),  
 decoration:  
 BoxDecoration(  
 border: Border.all(  
 color:  
 textBlackColor),  
 borderRadius:  
 BorderRadius  
 .all(  
 Radius.circular(  
 10.px),  
 )),  
 padding: EdgeInsets  
 .symmetric(  
 vertical:  
 1.h),  
 child:  
 getCustomTextW6S15(  
 text: 'Male'.tr,  
 color:  
 textBlackColor,  
 ),  
 ),  
 ),  
 GestureDetector(  
 onTap: () {  
 genderController  
 .text =  
 'Female'  
 .toString()  
 .tr;  
 Get.back();  
 },  
 child: Container(  
 width: 100.w,  
 alignment: Alignment  
 .center,  
 margin: EdgeInsets  
 .symmetric(  
 horizontal:  
 3.h),  
 decoration:  
 BoxDecoration(  
 border: Border.all(  
 color:  
 textBlackColor),  
 borderRadius:  
 BorderRadius  
 .all(  
 Radius.circular(  
 10.px),  
 )),  
 padding: EdgeInsets  
 .symmetric(  
 vertical:  
 1.h),  
 child:  
 getCustomTextW6S15(  
 text: 'Female'.tr,  
 color:  
 textBlackColor,  
 ),  
 ),  
 ),  
 GestureDetector(  
 onTap: () {  
 genderController  
 .text =  
 'Other'  
 .toString()  
 .tr;  
 Get.back();  
 },  
 child: Container(  
 width: 100.w,  
 alignment: Alignment  
 .center,  
 margin: EdgeInsets  
 .symmetric(  
 horizontal:  
 3.h),  
 decoration:  
 BoxDecoration(  
 border: Border.all(  
 color:  
 textBlackColor),  
 borderRadius:  
 BorderRadius  
 .all(  
 Radius.circular(  
 10.px),  
 )),  
 padding: EdgeInsets  
 .symmetric(  
 vertical:  
 1.h),  
 child:  
 getCustomTextW6S15(  
 text: 'Other'.tr,  
 color:  
 textBlackColor,  
 ),  
 ),  
 ),  
 ],  
 ),  
 );  
 },  
 );  
 },  
 hintText: 'Gender'.tr,  
 controller: genderController,  
 validator: (value) {  
 if (value!.isEmpty) {  
 return 'Gender not be empty!'  
 .tr;  
 } else {  
 return null;  
 }  
 },  
 ))  
 ],  
 ),  
 getVerSpace(2.h),  
 ],  
 ),  
 isActive: currentStep.value >= 1,  
 state: currentStep.value == 1  
 ? StepState.editing  
 : currentStep.value >= 1  
 ? StepState.complete  
 : StepState.disabled,  
 ),  
  
 //step 3  
 Step(  
 title: const Text(''),  
 content: Column(  
 children: <Widget>[  
 getCustomTextW6S15(  
 text:  
 'Please provide your Mobile Phone Number and Email to continue'  
 .tr,  
 color: textBlackColor,  
 textAlign: TextAlign.center),  
 getVerSpace(2.h),  
 InternationalPhoneNumberInput(  
 onInputChanged: (PhoneNumber number) {  
 log(number.phoneNumber.toString());  
 },  
 onInputValidated: (bool value) {  
 log(value.toString());  
 },  
 selectorConfig: const SelectorConfig(  
 selectorType:  
 PhoneInputSelectorType.DIALOG,  
 ),  
 ignoreBlank: false,  
 autoValidateMode: AutovalidateMode.disabled,  
 selectorTextStyle:  
 const TextStyle(color: Colors.black),  
 initialValue: number,  
 textFieldController: phoneNumberController,  
 formatInput: true,  
 keyboardType:  
 const TextInputType.numberWithOptions(  
 signed: true, decimal: true),  
 isEnabled: true,  
 inputDecoration: InputDecoration(  
 enabledBorder: UnderlineInputBorder(  
 borderSide: BorderSide(  
 color: backgroundColor,  
 width: 1.0.px),  
 borderRadius:  
 BorderRadius.circular(10.0.px)),  
 focusedBorder: UnderlineInputBorder(  
 borderSide: BorderSide(  
 color: backgroundColor,  
 width: 1.0.px),  
 borderRadius:  
 BorderRadius.circular(10.0.px)),  
 border: UnderlineInputBorder(  
 borderRadius:  
 BorderRadius.circular(10.0.px)),  
 fillColor: backgroundColor,  
 hintText: 'Phone Number'.tr,  
 hintStyle: getCustomTextStyleW5S15(  
 color: textBlackColor,  
 ),  
 enabled: true,  
 ),  
 cursorColor: textBlackColor,  
 onSaved: (PhoneNumber number) {  
 log('On Saved: $number');  
 },  
 ),  
 Divider(color: subTextColor),  
 Row(  
 children: [  
 Expanded(  
 flex: 1,  
 child: getCustomTextW6S15(  
 text: 'Email'.tr,  
 color: subTextColor),  
 ),  
 getHorSpace(2.h),  
 Expanded(  
 flex: 2,  
 child: getCustomTextFormField(  
 keyboardType:  
 TextInputType.emailAddress,  
 obscureText: false,  
 hintText: 'Enter Your Email'.tr,  
 controller: emailController,  
 validator: (value) => EmailValidator  
 .validate(value!)  
 ? null  
 : 'Please enter a valid email'  
 .tr,  
 ))  
 ],  
 ),  
 getVerSpace(2.h),  
 ],  
 ),  
 isActive: currentStep.value >= 2,  
 state: currentStep.value == 2  
 ? StepState.editing  
 : currentStep.value >= 2  
 ? StepState.complete  
 : StepState.disabled,  
 ),  
  
 Step(  
 title: const Text(''),  
 content: SingleChildScrollView(  
 child: Column(  
 children: <Widget>[  
 // getCustomTextW6S15(  
 // text:  
 // 'Please provide your Mobile Phone Number and Email to continue'.tr,  
 // color: textBlackColor,  
 // textAlign: TextAlign.center),  
 // getVerSpace(2.h),  
 // InternationalPhoneNumberInput(  
 // onInputChanged: (PhoneNumber number) {  
 // log(number.phoneNumber.toString());  
 // },  
 // onInputValidated: (bool value) {  
 // log(value.toString());  
 // },  
 // selectorConfig: const SelectorConfig(  
 // selectorType:  
 // PhoneInputSelectorType.DIALOG,  
 // ),  
 // ignoreBlank: false,  
 // autoValidateMode:  
 // AutovalidateMode.disabled,  
 // selectorTextStyle:  
 // const TextStyle(color: Colors.black),  
 // initialValue: number,  
 // textFieldController:  
 // phoneNumberController,  
 // formatInput: true,  
 // keyboardType:  
 // const TextInputType.numberWithOptions(  
 // signed: true, decimal: true),  
 // isEnabled: true,  
 // inputDecoration: InputDecoration(  
 // enabledBorder: UnderlineInputBorder(  
 // borderSide: BorderSide(  
 // color: backgroundColor,  
 // width: 1.0.px),  
 // borderRadius:  
 // BorderRadius.circular(10.0.px)),  
 // focusedBorder: UnderlineInputBorder(  
 // borderSide: BorderSide(  
 // color: backgroundColor,  
 // width: 1.0.px),  
 // borderRadius:  
 // BorderRadius.circular(10.0.px)),  
 // border: UnderlineInputBorder(  
 // borderRadius:  
 // BorderRadius.circular(10.0.px)),  
 // fillColor: backgroundColor,  
 // hintText: 'Phone Number'.tr,  
 // hintStyle: getCustomTextStyleW5S15(  
 // color: textBlackColor,  
 // ),  
 // enabled: true,  
 // ),  
 // cursorColor: textBlackColor,  
 // onSaved: (PhoneNumber number) {  
 // log('On Saved: $number');  
 // },  
 // ),  
 // Divider(color: subTextColor),  
 // Row(  
 // children: [  
 // Expanded(  
 // flex: 1,  
 // child: getCustomTextW6S15(  
 // text: 'Email'.tr,  
 // color: subTextColor),  
 // ),  
 // Expanded(  
 // flex: 2,  
 // child: getCustomTextFormField(  
 // keyboardType:  
 // TextInputType.emailAddress,  
 // obscureText: false,  
 // hintText: 'Enter Your Email'.tr,  
 // controller: emailController,  
 // validator: (value) => EmailValidator  
 // .validate(value!)  
 // ? null  
 // : 'Please enter a valid email'  
 // .tr,  
 // )),  
 // ],  
 // ),  
 // ElevatedButton(onPressed: null, child: Text('Next', style: TextStyle(color: Color(purple), backgroundColor: Color(pink)),)),  
  
 // SizedBox(height: 20,),  
  
 // GestureDetector(  
 // child: IgnorePointer(  
 // child: fillColorButton(  
 // color: Color(pink),  
 // text: 'Next'.tr,  
 // ),  
 // ),  
 // ),  
  
 SizedBox(  
 height: 20,  
 ),  
  
 getCustomTextW6S17(  
 text: 'Please enter the OTP Code'.tr,  
 color: textBlackColor,  
 textAlign: TextAlign.center),  
 getVerSpace(8.h),  
 PinCodeTextField(  
 length: 4,  
 keyboardType: TextInputType.phone,  
 appContext: context,  
 obscureText: false,  
 animationType: AnimationType.fade,  
 pinTheme: PinTheme(  
 borderWidth: 1.px,  
 shape: PinCodeFieldShape.box,  
 borderRadius:  
 BorderRadius.circular(5),  
 selectedColor: pinkAppColor,  
 selectedFillColor: backgroundColor,  
 inactiveColor: subTextColor,  
 inactiveFillColor: backgroundColor,  
 fieldHeight: 4.5.h,  
 fieldWidth: 4.h,  
 activeColor: pinkAppColor,  
 activeFillColor: backgroundColor,  
 fieldOuterPadding:  
 EdgeInsets.symmetric(  
 horizontal: 1.1.h)),  
 animationDuration:  
 const Duration(milliseconds: 200),  
 backgroundColor: backgroundColor,  
 mainAxisAlignment:  
 MainAxisAlignment.center,  
 enableActiveFill: true,  
 errorAnimationController: errorController,  
 controller: otpController,  
 onCompleted: (v) {  
 Get.to(() => WellDoneScreen(  
 screenTitle: 'otp',  
 ));  
 log("Completed\n OTP:${otpController.text}");  
 },  
 onChanged: (value) {},  
 beforeTextPaste: (text) {  
 log("Allowing to paste $text");  
 return true;  
 },  
 ),  
 getVerSpace(2.h),  
 getCustomTextW6S15(  
 text: 'I did not receive the OTP'.tr,  
 color: subTextColor),  
 getVerSpace(1.h),  
 getCustomTextW6S15(  
 text: 'Resend Code'.tr,  
 color: pinkAppColor),  
 getVerSpace(2.h),  
 ],  
 ),  
 ),  
 isActive: currentStep.value >= 3,  
 state: currentStep.value == 3  
 ? StepState.editing  
 : currentStep.value >= 3  
 ? StepState.complete  
 : StepState.disabled,  
 ),  
 ],  
 ),  
 ),  
 ),  
 ),  
 ],  
 ),  
 ),  
 ),  
 ),  
 ),  
 );  
 }  
  
 tapped(int step) {  
 currentStep.value = step;  
 }  
  
 continued() {  
 currentStep.value < 3 ? currentStep.value += 1 : null;  
 }  
  
 cancel() {  
 currentStep.value > 0 ? currentStep.value -= 1 : null;  
 }  
}

# AuthScreens

## scan\_id\_shop\_owner\_screen

import 'dart:async';  
import 'dart:developer';  
  
import 'package:algorthimi/utils/app\_color.dart';  
import 'package:algorthimi/view/AuthScreens/scan\_id\_camera\_screen.dart';  
import 'package:algorthimi/view/AuthScreens/scan\_id\_camera\_shop\_owner\_screen.dart';  
import 'package:algorthimi/view/otp\_screen.dart';  
import 'package:email\_validator/email\_validator.dart';  
import 'package:flutter/cupertino.dart';  
import 'package:flutter/material.dart';  
import 'package:get/get.dart';  
import 'package:intl\_phone\_number\_input/intl\_phone\_number\_input.dart';  
import 'package:pin\_code\_fields/pin\_code\_fields.dart';  
import 'package:responsive\_sizer/responsive\_sizer.dart';  
  
import '../../utils/color\_data.dart';  
import '../Widgets/widget\_utils.dart';  
import 'welldone\_screen.dart';  
  
class ScanIdShopOwnerScreen extends StatefulWidget {  
 const ScanIdShopOwnerScreen({Key? key}) : super(key: key);  
  
 @override  
 State<ScanIdShopOwnerScreen> createState() => \_ScanIdShopOwnerScreenState();  
}  
  
class \_ScanIdShopOwnerScreenState extends State<ScanIdShopOwnerScreen> {  
 TextEditingController idNumberController = TextEditingController();  
 TextEditingController cardNumberController = TextEditingController();  
 TextEditingController fullNameController = TextEditingController();  
 TextEditingController dateOfBrithController = TextEditingController();  
 TextEditingController nationalityController = TextEditingController();  
 TextEditingController genderController = TextEditingController();  
 TextEditingController expiryDateController = TextEditingController();  
 TextEditingController phoneNumberController = TextEditingController();  
 TextEditingController emailController = TextEditingController();  
 TextEditingController licenseNoController = TextEditingController();  
 TextEditingController tradeNameController = TextEditingController();  
 TextEditingController tradeExpiryDateController = TextEditingController();  
 TextEditingController otpController = TextEditingController();  
 StreamController<ErrorAnimationType>? errorController;  
 String initialCountry = 'AE';  
 PhoneNumber number = PhoneNumber(isoCode: 'AE');  
  
 var currentStep = 3.obs;  
 StepperType stepperType = StepperType.horizontal;  
  
 back() {  
 isScannedFreelancerId.value = false;  
 isScannedShopOwnerId.value = false;  
 isScannedShopOwnerLicense.value = false;  
 Get.back();  
 // Get.back();  
 }  
  
 @override  
 Widget build(BuildContext context) {  
 return Obx(  
 () => WillPopScope(  
 onWillPop: () {  
 return back();  
 },  
 child: Scaffold(  
 resizeToAvoidBottomInset: false,  
 backgroundColor: backgroundColor,  
 body: SizedBox(  
 height: 100.h,  
 width: 100.h,  
 child: Stack(  
 children: [  
 Container(  
 height: 22.h,  
 width: 100.w,  
 alignment: Alignment.topCenter,  
 padding: EdgeInsets.only(top: 6.h, left: 3.h, right: 3.h),  
 decoration: BoxDecoration(  
 color: accentColor,  
 borderRadius:  
 BorderRadius.only(bottomRight: Radius.circular(4.h))),  
 child: Row(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 mainAxisAlignment: MainAxisAlignment.start,  
 children: [  
 GestureDetector(  
 onTap: () {  
 isScannedFreelancerId.value = false;  
 isScannedShopOwnerId.value = false;  
 isScannedShopOwnerLicense.value = false;  
 Get.back();  
 // Get.back();  
 },  
 child: Icon(Icons.arrow\_back\_ios\_new\_outlined,  
 color: backgroundColor, size: 2.5.h),  
 ),  
 const Spacer(),  
 SizedBox(  
 width: 30.h,  
 child: getCustomTextW6S22(  
 text: currentStep.value == 0 &&  
 !isScannedShopOwnerId.value  
 ? 'Scan Your ID'.tr  
 : currentStep.value == 0 ||  
 currentStep.value == 1  
 ? 'Personal Details\nConfirmation'.tr  
 : currentStep.value == 2  
 ? 'Mobile & Email\nConfirmation'.tr  
 : currentStep.value == 3 &&  
 !isScannedShopOwnerLicense.value  
 ? 'Scan Your License'.tr  
 : currentStep.value == 3  
 ? 'License Details\nConfirmation'  
 .tr  
 : 'Enter Verification\nCode'.tr,  
 color: textWhiteColor,  
 textAlign: TextAlign.center),  
 ),  
 const Spacer()  
 ],  
 ),  
 ),  
 Positioned(  
 top: 16.h,  
 right: 0,  
 left: 0,  
 child: Container(  
 width: 100.h,  
 height: 100.h,  
 padding: EdgeInsets.only(top: 2.h),  
 alignment: Alignment.topCenter,  
 margin: EdgeInsets.symmetric(horizontal: 3.h),  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius: BorderRadius.all(Radius.circular(10.px))),  
 child: Theme(  
 data: ThemeData(  
 canvasColor: backgroundColor,  
 colorScheme: Theme.of(context)  
 .colorScheme  
 .copyWith(primary: accentColor),  
 ),  
 child: Stepper(  
 type: stepperType,  
 elevation: 0,  
 physics: const NeverScrollableScrollPhysics(),  
 currentStep: currentStep.value,  
 onStepTapped: (step) => tapped(step),  
 onStepContinue: continued,  
 onStepCancel: cancel,  
 controlsBuilder:  
 (BuildContext context, ControlsDetails details) {  
 return Obx(  
 () => isScannedShopOwnerId.value  
 ? currentStep.value == 4  
 ? Container()  
 : currentStep.value != 3 ||  
 isScannedShopOwnerLicense.value ==  
 true  
 ? Row(children: [  
 Expanded(  
 child: ElevatedButton(  
 style: ButtonStyle(  
 backgroundColor:  
 MaterialStatePropertyAll(  
 pinkAppColor),  
 overlayColor:  
 MaterialStatePropertyAll(  
 pinkAppColor.withOpacity(  
 0.1)),  
 side: MaterialStateProperty.all(  
 BorderSide(  
 color: pinkAppColor,  
 width: 1.5,  
 style: BorderStyle  
 .solid)),  
 shape: MaterialStatePropertyAll(  
 RoundedRectangleBorder(  
 borderRadius:  
 BorderRadius.all(  
 Radius.circular(10.px)))),  
 padding: MaterialStatePropertyAll(EdgeInsets.symmetric(vertical: 1.6.h))),  
 onPressed:(){ Get.to(() => const OtpScreen());},  
 // onPressed: details.onStepContinue,  
 child: getCustomTextW6S16(  
 maxLines: 1,  
 text: currentStep.value == 3  
 ? 'Confirm'.tr  
 : 'Next'.tr,  
 color: textWhiteColor,  
 ),  
 )),  
 if (currentStep.value != 0)  
 Expanded(  
 child: Row(  
 children: [  
 getHorSpace(2.h),  
 Expanded(  
 child: OutlinedButton(  
 style: ButtonStyle(  
 overlayColor: MaterialStatePropertyAll(  
 pinkAppColor  
 .withOpacity(  
 0.1)),  
 side: MaterialStateProperty  
 .all(BorderSide(  
 color:  
 pinkAppColor,  
 width:  
 1.5,  
 style: BorderStyle  
 .solid)),  
 shape: MaterialStatePropertyAll(  
 RoundedRectangleBorder(  
 borderRadius:  
 BorderRadius.all(Radius.circular(10.px)))),  
 padding: MaterialStatePropertyAll(EdgeInsets.symmetric(vertical: 1.6.h))),  
 onPressed: details.onStepCancel,  
 child: getCustomTextW6S16(text: 'Back'.tr, color: pinkAppColor)),  
 ),  
 ],  
 ),  
 )  
 ])  
 : Container()  
 : Container(),  
 );  
 },  
 steps: <Step>[  
 Step(  
 title: const Text(''),  
 content: isScannedShopOwnerId.value  
 ? Column(  
 children: [  
 getCustomTextW6S15(  
 text:  
 'Please confirm your\nPersonal Information'  
 .tr,  
 color: textBlackColor,  
 textAlign: TextAlign.center),  
 getVerSpace(2.h),  
 Row(  
 children: [  
 Expanded(  
 child: getCustomTextW6S15(  
 text: 'ID Number'.tr,  
 color: subTextColor),  
 ),  
 Expanded(  
 flex: 2,  
 child: getCustomTextFormField(  
 keyboardType:  
 const TextInputType  
 .numberWithOptions(  
 signed: true,  
 decimal: true),  
 obscureText: false,  
 hintText: 'ID Number'.tr,  
 controller: idNumberController,  
 validator: (value) {  
 if (value!.isEmpty) {  
 return 'ID Number not be empty!'  
 .tr;  
 } else {  
 return null;  
 }  
 },  
 ))  
 ],  
 ),  
 Divider(color: subTextColor),  
 Row(  
 children: [  
 Expanded(  
 child: getCustomTextW6S15(  
 text: 'Full Name'.tr,  
 color: subTextColor),  
 ),  
 Expanded(  
 flex: 2,  
 child: getCustomTextFormField(  
 keyboardType:  
 TextInputType.text,  
 obscureText: false,  
 hintText: 'Full Name'.tr,  
 controller: fullNameController,  
 validator: (value) {  
 if (value!.isEmpty) {  
 return 'Full Name not be empty!'  
 .tr;  
 } else {  
 return null;  
 }  
 },  
 ))  
 ],  
 ),  
 Divider(color: subTextColor),  
 Row(  
 children: [  
 Expanded(  
 child: getCustomTextW6S15(  
 text: 'Nationality'.tr,  
 color: subTextColor),  
 ),  
 Expanded(  
 flex: 2,  
 child: getCustomTextFormField(  
 keyboardType:  
 TextInputType.text,  
 obscureText: false,  
 hintText: 'Nationality'.tr,  
 controller:  
 nationalityController,  
 validator: (value) {  
 if (value!.isEmpty) {  
 return 'Nationality not be empty!'  
 .tr;  
 } else {  
 return null;  
 }  
 },  
 ))  
 ],  
 ),  
 getVerSpace(2.h),  
 ],  
 )  
 : Column(  
 mainAxisAlignment: MainAxisAlignment.start,  
 crossAxisAlignment:  
 CrossAxisAlignment.center,  
 children: <Widget>[  
 RichText(  
 textAlign: TextAlign.center,  
 text: TextSpan(  
 text: 'Scan the back of your '.tr,  
 style: getCustomTextStyleW5S15(  
 color: textBlackColor,  
 ),  
 children: [  
 TextSpan(  
 text: 'Emirates ID'.tr,  
 style: getCustomTextStyleW7S15(  
 color: textBlackColor,  
 ),  
 ),  
 ]),  
 ),  
 getVerSpace(10.h),  
 getSvgImage('scan\_id.svg', height: 23.h),  
 getVerSpace(10.h),  
 GestureDetector(  
 onTap: () {  
 Get.to(  
 () => const ScanIdCameraScreen());  
 },  
 child: fillColorButton(  
 color: pinkAppColor,  
 text: 'Start Scanning'.tr,  
 ),  
 ),  
 getVerSpace(2.h),  
 ],  
 ),  
 isActive: currentStep.value >= 0,  
 state: currentStep.value == 0  
 ? StepState.editing  
 : StepState.complete,  
 ),  
 Step(  
 title: const Text(''),  
 content: Column(  
 children: <Widget>[  
 getCustomTextW6S15(  
 text:  
 'Please confirm your\nPersonal Information'  
 .tr,  
 color: textBlackColor,  
 textAlign: TextAlign.center),  
 getVerSpace(2.h),  
 Row(  
 children: [  
 Expanded(  
 child: getCustomTextW6S15(  
 text: 'Card Number'.tr,  
 color: subTextColor),  
 ),  
 Expanded(  
 flex: 2,  
 child: getCustomTextFormField(  
 keyboardType: const TextInputType  
 .numberWithOptions(  
 signed: true, decimal: true),  
 obscureText: false,  
 hintText: 'Card Number'.tr,  
 controller: cardNumberController,  
 validator: (value) {  
 if (value!.isEmpty) {  
 return 'Card Number not be empty!'  
 .tr;  
 } else {  
 return null;  
 }  
 },  
 ))  
 ],  
 ),  
 Divider(color: subTextColor),  
 Row(  
 children: [  
 Expanded(  
 child: getCustomTextW6S15(  
 text: 'Date of Birth'.tr,  
 color: subTextColor),  
 ),  
 Expanded(  
 flex: 2,  
 child: getCustomTextFormField(  
 obscureText: false,  
 readOnly: true,  
 onTap: () async {  
 await showCupertinoModalPopup<void>(  
 context: context,  
 builder: (\_) {  
 return Container(  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius:  
 BorderRadius.only(  
 topLeft: Radius.circular(  
 12.px),  
 topRight: Radius.circular(  
 12.px),  
 ),  
 ),  
 height: 25.h,  
 child: CupertinoDatePicker(  
 mode:  
 CupertinoDatePickerMode  
 .date,  
 onDateTimeChanged: (value) {  
 dateOfBrithController  
 .text =  
 value  
 .toString()  
 .substring(0, 10);  
 },  
 ),  
 );  
 },  
 );  
 },  
 hintText: 'Date of Birth'.tr,  
 controller: dateOfBrithController,  
 validator: (value) {  
 if (value!.isEmpty) {  
 return 'Date of Birth not be empty!'  
 .tr;  
 } else {  
 return null;  
 }  
 },  
 ))  
 ],  
 ),  
 Divider(color: subTextColor),  
 Row(  
 children: [  
 Expanded(  
 child: getCustomTextW6S15(  
 text: 'Expiry Date'.tr,  
 color: subTextColor),  
 ),  
 Expanded(  
 flex: 2,  
 child: getCustomTextFormField(  
 obscureText: false,  
 readOnly: true,  
 onTap: () async {  
 await showCupertinoModalPopup<void>(  
 context: context,  
 builder: (\_) {  
 return Container(  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius:  
 BorderRadius.only(  
 topLeft: Radius.circular(  
 12.px),  
 topRight: Radius.circular(  
 12.px),  
 ),  
 ),  
 height: 25.h,  
 child: CupertinoDatePicker(  
 mode:  
 CupertinoDatePickerMode  
 .date,  
 onDateTimeChanged: (value) {  
 expiryDateController  
 .text =  
 value  
 .toString()  
 .substring(0, 10);  
 },  
 ),  
 );  
 },  
 );  
 },  
 hintText: 'Expiry Date'.tr,  
 controller: expiryDateController,  
 validator: (value) {  
 if (value!.isEmpty) {  
 return 'Expiry Date not be empty!'  
 .tr;  
 } else {  
 return null;  
 }  
 },  
 ))  
 ],  
 ),  
 Divider(color: subTextColor),  
 Row(  
 children: [  
 Expanded(  
 child: getCustomTextW6S15(  
 text: 'Gender'.tr,  
 color: subTextColor),  
 ),  
 Expanded(  
 flex: 2,  
 child: getCustomTextFormField(  
 obscureText: false,  
 readOnly: true,  
 onTap: () {  
 showModalBottomSheet<void>(  
 context: context,  
 elevation: 5,  
 builder: (BuildContext context) {  
 return Container(  
 width: 100.w,  
 height: 25.h,  
 padding: EdgeInsets.symmetric(  
 vertical: 2.h),  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius:  
 BorderRadius.only(  
 topLeft: Radius  
 .circular(  
 15.px),  
 topRight: Radius  
 .circular(  
 15.px))),  
 child: Column(  
 crossAxisAlignment:  
 CrossAxisAlignment  
 .start,  
 mainAxisAlignment:  
 MainAxisAlignment  
 .spaceEvenly,  
 children: [  
 GestureDetector(  
 onTap: () {  
 genderController  
 .text =  
 'Male'  
 .toString()  
 .tr;  
 Get.back();  
 },  
 child: Container(  
 width: 100.w,  
 alignment:  
 Alignment.center,  
 margin: EdgeInsets  
 .symmetric(  
 horizontal:  
 3.h),  
 decoration:  
 BoxDecoration(  
 border: Border  
 .all(  
 color:  
 textBlackColor),  
 borderRadius:  
 BorderRadius  
 .all(  
 Radius.circular(  
 10.px),  
 )),  
 padding: EdgeInsets  
 .symmetric(  
 vertical:  
 1.h),  
 child:  
 getCustomTextW6S15(  
 text: 'Male'.tr,  
 color:  
 textBlackColor,  
 ),  
 ),  
 ),  
 GestureDetector(  
 onTap: () {  
 genderController  
 .text =  
 'Female'  
 .toString()  
 .tr;  
 Get.back();  
 },  
 child: Container(  
 width: 100.w,  
 alignment:  
 Alignment.center,  
 margin: EdgeInsets  
 .symmetric(  
 horizontal:  
 3.h),  
 decoration:  
 BoxDecoration(  
 border: Border  
 .all(  
 color:  
 textBlackColor),  
 borderRadius:  
 BorderRadius  
 .all(  
 Radius.circular(  
 10.px),  
 )),  
 padding: EdgeInsets  
 .symmetric(  
 vertical:  
 1.h),  
 child:  
 getCustomTextW6S15(  
 text: 'Female'.tr,  
 color:  
 textBlackColor,  
 ),  
 ),  
 ),  
 GestureDetector(  
 onTap: () {  
 genderController  
 .text =  
 'Other'  
 .toString()  
 .tr;  
 Get.back();  
 },  
 child: Container(  
 width: 100.w,  
 alignment:  
 Alignment.center,  
 margin: EdgeInsets  
 .symmetric(  
 horizontal:  
 3.h),  
 decoration:  
 BoxDecoration(  
 border: Border  
 .all(  
 color:  
 textBlackColor),  
 borderRadius:  
 BorderRadius  
 .all(  
 Radius.circular(  
 10.px),  
 )),  
 padding: EdgeInsets  
 .symmetric(  
 vertical:  
 1.h),  
 child:  
 getCustomTextW6S15(  
 text: 'Other'.tr,  
 color:  
 textBlackColor,  
 ),  
 ),  
 ),  
 ],  
 ),  
 );  
 },  
 );  
 },  
 hintText: 'Gender'.tr,  
 controller: genderController,  
 validator: (value) {  
 if (value!.isEmpty) {  
 return 'Gender not be empty!'.tr;  
 } else {  
 return null;  
 }  
 },  
 ))  
 ],  
 ),  
 getVerSpace(2.h),  
 ],  
 ),  
 isActive: currentStep.value >= 1,  
 state: currentStep.value == 1  
 ? StepState.editing  
 : currentStep.value >= 1  
 ? StepState.complete  
 : StepState.disabled,  
 ),  
 Step(  
 title: const Text(''),  
 content: Column(  
 children: <Widget>[  
 getCustomTextW6S15(  
 text:  
 'Please provide your Mobile Phone Number and Email to continue'  
 .tr,  
 color: textBlackColor,  
 textAlign: TextAlign.center),  
 getVerSpace(2.h),  
 InternationalPhoneNumberInput(  
 onInputChanged: (PhoneNumber number) {  
 log(number.phoneNumber.toString());  
 },  
 onInputValidated: (bool value) {  
 log(value.toString());  
 },  
 selectorConfig: const SelectorConfig(  
 selectorType: PhoneInputSelectorType.DIALOG,  
 ),  
 ignoreBlank: false,  
 autoValidateMode: AutovalidateMode.disabled,  
 selectorTextStyle:  
 const TextStyle(color: Colors.black),  
 initialValue: number,  
 textFieldController: phoneNumberController,  
 formatInput: true,  
 keyboardType:  
 const TextInputType.numberWithOptions(  
 signed: true, decimal: true),  
 isEnabled: true,  
 inputDecoration: InputDecoration(  
 enabledBorder: UnderlineInputBorder(  
 borderSide: BorderSide(  
 color: backgroundColor,  
 width: 1.0.px),  
 borderRadius:  
 BorderRadius.circular(10.0.px)),  
 focusedBorder: UnderlineInputBorder(  
 borderSide: BorderSide(  
 color: backgroundColor,  
 width: 1.0.px),  
 borderRadius:  
 BorderRadius.circular(10.0.px)),  
 border: UnderlineInputBorder(  
 borderRadius:  
 BorderRadius.circular(10.0.px)),  
 fillColor: backgroundColor,  
 hintText: 'Phone Number'.tr,  
 hintStyle: getCustomTextStyleW5S15(  
 color: textBlackColor,  
 ),  
 enabled: true,  
 ),  
 cursorColor: textBlackColor,  
 onSaved: (PhoneNumber number) {  
 log('On Saved: $number');  
 },  
 ),  
 Divider(color: subTextColor),  
 Row(  
 children: [  
 Expanded(  
 flex: 1,  
 child: getCustomTextW6S15(  
 text: 'Email'.tr,  
 color: subTextColor),  
 ),  
 getHorSpace(2.h),  
 Expanded(  
 flex: 2,  
 child: getCustomTextFormField(  
 keyboardType:  
 TextInputType.emailAddress,  
 obscureText: false,  
 hintText: 'Enter Your Email'.tr,  
 controller: emailController,  
 validator: (value) =>  
 EmailValidator.validate(value!)  
 ? null  
 : 'Please enter a valid email'  
 .tr,  
 ))  
 ],  
 ),  
 getVerSpace(2.h),  
 ],  
 ),  
 isActive: currentStep.value >= 2,  
 state: currentStep.value == 2  
 ? StepState.editing  
 : currentStep.value >= 2  
 ? StepState.complete  
 : StepState.disabled,  
 ),  
 Step(  
 title: const Text(''),  
 content: isScannedShopOwnerLicense.value  
 ? Column(  
 children: [  
 getCustomTextW6S15(  
 text:  
 'Please confirm your\nLicense Information'  
 .tr,  
 color: textBlackColor,  
 textAlign: TextAlign.center),  
 getVerSpace(2.h),  
 Row(  
 children: [  
 Expanded(  
 child: getCustomTextW6S15(  
 text: 'License No'.tr,  
 color: subTextColor),  
 ),  
 Expanded(  
 flex: 2,  
 child: getCustomTextFormField(  
 keyboardType:  
 const TextInputType  
 .numberWithOptions(  
 signed: true,  
 decimal: true),  
 obscureText: false,  
 hintText: 'License No'.tr,  
 controller: licenseNoController,  
 validator: (value) {  
 if (value!.isEmpty) {  
 return 'License No not be empty!'  
 .tr;  
 } else {  
 return null;  
 }  
 },  
 ))  
 ],  
 ),  
 Divider(color: subTextColor),  
 Row(  
 children: [  
 Expanded(  
 child: getCustomTextW6S15(  
 text: 'Trade Name'.tr,  
 color: subTextColor),  
 ),  
 Expanded(  
 flex: 2,  
 child: getCustomTextFormField(  
 keyboardType:  
 TextInputType.text,  
 obscureText: false,  
 hintText: 'Trade Name'.tr,  
 controller: tradeNameController,  
 validator: (value) {  
 if (value!.isEmpty) {  
 return 'Trade Name not be empty!'  
 .tr;  
 } else {  
 return null;  
 }  
 },  
 ))  
 ],  
 ),  
 Divider(color: subTextColor),  
 Row(  
 children: [  
 Expanded(  
 child: getCustomTextW6S15(  
 text: 'Expiry Date'.tr,  
 color: subTextColor),  
 ),  
 Expanded(  
 flex: 2,  
 child: getCustomTextFormField(  
 obscureText: false,  
 readOnly: true,  
 onTap: () async {  
 await showCupertinoModalPopup<  
 void>(  
 context: context,  
 builder: (\_) {  
 return Container(  
 decoration:  
 BoxDecoration(  
 color:  
 backgroundColor,  
 borderRadius:  
 BorderRadius.only(  
 topLeft:  
 Radius.circular(  
 12.px),  
 topRight:  
 Radius.circular(  
 12.px),  
 ),  
 ),  
 height: 25.h,  
 child:  
 CupertinoDatePicker(  
 mode:  
 CupertinoDatePickerMode  
 .date,  
 onDateTimeChanged:  
 (value) {  
 tradeExpiryDateController  
 .text =  
 value  
 .toString()  
 .substring(  
 0, 10);  
 },  
 ),  
 );  
 },  
 );  
 },  
 hintText: 'Expiry Date'.tr,  
 controller:  
 tradeExpiryDateController,  
 validator: (value) {  
 if (value!.isEmpty) {  
 return 'Expiry Date not be empty!'  
 .tr;  
 } else {  
 return null;  
 }  
 },  
 ))  
 ],  
 ),  
 getVerSpace(2.h),  
 ],  
 )  
 : Column(  
 mainAxisAlignment: MainAxisAlignment.start,  
 crossAxisAlignment:  
 CrossAxisAlignment.center,  
 children: <Widget>[  
 RichText(  
 textAlign: TextAlign.center,  
 text: TextSpan(  
 text: 'Scan the back of your\n'.tr,  
 style: getCustomTextStyleW5S15(  
 color: textBlackColor,  
 ),  
 children: [  
 TextSpan(  
 text: 'Trade License'.tr,  
 style: getCustomTextStyleW7S15(  
 color: textBlackColor,  
 ),  
 ),  
 ]),  
 ),  
 getVerSpace(10.h),  
 getSvgImage('scan\_id\_shop\_owner.svg',  
 height: 25.h),  
 getVerSpace(10.h),  
 GestureDetector(  
 onTap: () {  
 Get.to(() =>  
 const ScanIdCameraShopOwnerScreen());  
 },  
 child: fillColorButton(  
 color: pinkAppColor,  
 text: 'Start Scanning'.tr,  
 ),  
 ),  
 getVerSpace(2.h),  
 ],  
 ),  
 isActive: currentStep.value >= 3,  
 state: currentStep.value == 3  
 ? StepState.editing  
 : currentStep.value >= 3  
 ? StepState.complete  
 : StepState.disabled,  
 ),  
  
 //step 5  
  
 // Step(  
 // title: const Text(''),  
 // content: Column(  
 // children: <Widget>[  
 // getCustomTextW6S15(  
 // text:  
 // 'Please enter the OTP Code\nwe sent you'.tr,  
 // color: textBlackColor,  
 // textAlign: TextAlign.center),  
 // getVerSpace(4.h),  
 // PinCodeTextField(  
 // length: 4,  
 // keyboardType: TextInputType.phone,  
 // appContext: context,  
 // obscureText: false,  
 // animationType: AnimationType.fade,  
 // pinTheme: PinTheme(  
 // borderWidth: 1.px,  
 // shape: PinCodeFieldShape.box,  
 // borderRadius: BorderRadius.circular(5),  
 // selectedColor: pinkAppColor,  
 // selectedFillColor: backgroundColor,  
 // inactiveColor: subTextColor,  
 // inactiveFillColor: backgroundColor,  
 // fieldHeight: 5.5.h,  
 // fieldWidth: 4.h,  
 // activeColor: pinkAppColor,  
 // activeFillColor: backgroundColor,  
 // fieldOuterPadding: EdgeInsets.symmetric(  
 // horizontal: 1.1.h)),  
 // animationDuration:  
 // const Duration(milliseconds: 200),  
 // backgroundColor: backgroundColor,  
 // mainAxisAlignment: MainAxisAlignment.center,  
 // enableActiveFill: true,  
 // errorAnimationController: errorController,  
 // controller: otpController,  
 // onCompleted: (v) {  
 // isScannedFreelancerId.value = false;  
 // isScannedShopOwnerId.value = false;  
 // isScannedShopOwnerLicense.value = false;  
 // Get.to(() => WellDoneScreen(  
 // screenTitle: 'otpLicense',  
 // ));  
 // log("Completed\n OTP:${otpController.text}");  
 // },  
 // onChanged: (value) {},  
 // beforeTextPaste: (text) {  
 // log("Allowing to paste $text");  
 // return true;  
 // },  
 // ),  
 // getVerSpace(2.h),  
 // getCustomTextW6S15(  
 // text: 'I did not receive the OTP'.tr,  
 // color: subTextColor),  
 // getVerSpace(1.8.h),  
 // getCustomTextW6S15(  
 // text: 'Resend Code'.tr,  
 // color: pinkAppColor),  
 // getVerSpace(2.h),  
 // ],  
 // ),  
 // isActive: currentStep.value >= 4,  
 // state: currentStep.value == 4  
 // ? StepState.editing  
 // : currentStep.value >= 4  
 // ? StepState.complete  
 // : StepState.disabled,  
 // ),  
  
 Step(  
 title: const Text(''),  
 content: SingleChildScrollView(  
 child: Column(  
 children: <Widget>[  
 // getCustomTextW6S15(  
 // text:  
 // 'Please provide your Mobile Phone Number and Email to continue'.tr,  
 // color: textBlackColor,  
 // textAlign: TextAlign.center),  
 // getVerSpace(2.h),  
 InternationalPhoneNumberInput(  
 onInputChanged: (PhoneNumber number) {  
 log(number.phoneNumber.toString());  
 },  
 onInputValidated: (bool value) {  
 log(value.toString());  
 },  
 selectorConfig: const SelectorConfig(  
 selectorType:  
 PhoneInputSelectorType.DIALOG,  
 ),  
 ignoreBlank: false,  
 autoValidateMode: AutovalidateMode.disabled,  
 selectorTextStyle:  
 const TextStyle(color: Colors.black),  
 initialValue: number,  
 textFieldController: phoneNumberController,  
 formatInput: true,  
 keyboardType:  
 const TextInputType.numberWithOptions(  
 signed: true, decimal: true),  
 isEnabled: true,  
 inputDecoration: InputDecoration(  
 enabledBorder: UnderlineInputBorder(  
 borderSide: BorderSide(  
 color: backgroundColor,  
 width: 1.0.px),  
 borderRadius:  
 BorderRadius.circular(10.0.px)),  
 focusedBorder: UnderlineInputBorder(  
 borderSide: BorderSide(  
 color: backgroundColor,  
 width: 1.0.px),  
 borderRadius:  
 BorderRadius.circular(10.0.px)),  
 border: UnderlineInputBorder(  
 borderRadius:  
 BorderRadius.circular(10.0.px)),  
 fillColor: backgroundColor,  
 hintText: 'Phone Number'.tr,  
 hintStyle: getCustomTextStyleW5S15(  
 color: textBlackColor,  
 ),  
 enabled: true,  
 ),  
 cursorColor: textBlackColor,  
 onSaved: (PhoneNumber number) {  
 log('On Saved: $number');  
 },  
 ),  
 Divider(color: subTextColor),  
 Row(  
 children: [  
 Expanded(  
 flex: 1,  
 child: getCustomTextW6S15(  
 text: 'Email'.tr,  
 color: subTextColor),  
 ),  
 Expanded(  
 flex: 2,  
 child: getCustomTextFormField(  
 keyboardType:  
 TextInputType.emailAddress,  
 obscureText: false,  
 hintText: 'Enter Your Email'.tr,  
 controller: emailController,  
 validator: (value) => EmailValidator  
 .validate(value!)  
 ? null  
 : 'Please enter a valid email'  
 .tr,  
 )),  
 ],  
 ),  
 // ElevatedButton(onPressed: null, child: Text('Next', style: TextStyle(color: Color(purple), backgroundColor: Color(pink)),)),  
  
 // SizedBox(height: 20,),  
  
 GestureDetector(  
 child: IgnorePointer(  
 child: fillColorButton(  
 color: Color(pink),  
 text: 'Next'.tr,  
 ),  
 ),  
 ),  
  
 SizedBox(  
 height: 15,  
 ),  
  
 getCustomTextW6S15(  
 text: 'Please enter the OTP Code'.tr,  
 color: textBlackColor,  
 textAlign: TextAlign.center),  
 getVerSpace(2.h),  
 PinCodeTextField(  
 length: 4,  
 keyboardType: TextInputType.phone,  
 appContext: context,  
 obscureText: false,  
 animationType: AnimationType.fade,  
 pinTheme: PinTheme(  
 borderWidth: 1.px,  
 shape: PinCodeFieldShape.box,  
 borderRadius: BorderRadius.circular(5),  
 selectedColor: pinkAppColor,  
 selectedFillColor: backgroundColor,  
 inactiveColor: subTextColor,  
 inactiveFillColor: backgroundColor,  
 fieldHeight: 4.5.h,  
 fieldWidth: 4.h,  
 activeColor: pinkAppColor,  
 activeFillColor: backgroundColor,  
 fieldOuterPadding: EdgeInsets.symmetric(  
 horizontal: 1.1.h)),  
 animationDuration:  
 const Duration(milliseconds: 200),  
 backgroundColor: backgroundColor,  
 mainAxisAlignment: MainAxisAlignment.center,  
 enableActiveFill: true,  
 errorAnimationController: errorController,  
 controller: otpController,  
 onCompleted: (v) {  
 Get.to(() => WellDoneScreen(  
 screenTitle: 'otp',  
 ));  
 log("Completed\n OTP:${otpController.text}");  
 },  
 onChanged: (value) {},  
 beforeTextPaste: (text) {  
 log("Allowing to paste $text");  
 return true;  
 },  
 ),  
 // getVerSpace(2.h),  
 getCustomTextW6S15(  
 text: 'I did not receive the OTP'.tr,  
 color: subTextColor),  
 getVerSpace(1.h),  
 getCustomTextW6S15(  
 text: 'Resend Code'.tr,  
 color: pinkAppColor),  
 getVerSpace(2.h),  
 ],  
 ),  
 ),  
  
 isActive: currentStep.value >= 4,  
 state: currentStep.value == 4  
 ? StepState.editing  
 : currentStep.value >= 4  
 ? StepState.complete  
 : StepState.disabled,  
  
 // isActive: currentStep.value >= 3,  
 // state: currentStep.value == 3  
 // ? StepState.editing  
 // : currentStep.value >= 3  
 // ? StepState.complete  
 // : StepState.disabled,  
 ),  
 ],  
 ),  
 ),  
 ),  
 ),  
 ],  
 ),  
 ),  
 ),  
 ),  
 );  
 }  
  
 tapped(int step) {  
 currentStep.value = step;  
 }  
  
 continued() {  
 currentStep.value < 4 ? currentStep.value += 1 : null;  
 }  
  
 cancel() {  
 currentStep.value > 0 ? currentStep.value -= 1 : null;  
 }  
}

# AuthScreens

## sign\_up\_screen

import 'dart:async';  
  
import 'package:algorthimi/view/AuthScreens/login\_screen.dart';  
import 'package:algorthimi/view/AuthScreens/take\_picture\_screen.dart';  
import 'package:flutter/material.dart';  
import 'package:get/get.dart';  
import 'package:responsive\_sizer/responsive\_sizer.dart';  
  
import '../../utils/color\_data.dart';  
import '../Widgets/widget\_utils.dart';  
  
class SignUpScreen extends StatefulWidget {  
 const SignUpScreen({Key? key}) : super(key: key);  
  
 @override  
 State<SignUpScreen> createState() => \_SignUpScreenState();  
}  
  
class \_SignUpScreenState extends State<SignUpScreen> {  
 var isSelectSignUp = 100.obs;  
  
 @override  
 Widget build(BuildContext context) {  
 return Obx(  
 () => Scaffold(  
 resizeToAvoidBottomInset: false,  
 backgroundColor: backgroundColor,  
 body: Container(  
 width: 100.h,  
 height: 100.h - 10.h,  
 padding:  
 EdgeInsets.only(top: 16.h, right: 3.h, left: 3.h, bottom: 10.h),  
 decoration: BoxDecoration(  
 color: accentColor,  
 borderRadius:  
 BorderRadius.only(bottomRight: Radius.circular(20.h))),  
 child: Column(  
 mainAxisAlignment: MainAxisAlignment.spaceBetween,  
 crossAxisAlignment: CrossAxisAlignment.center,  
 children: [  
 Column(  
 children: [  
 Align(  
 alignment: Alignment.center,  
 child: getSvgImage('app\_logo.svg', height: 12.h)),  
 getVerSpace(1.h),  
 getCustomTextW6S22(text: 'Sign Up'.tr, color: textWhiteColor),  
 getVerSpace(2.h),  
 getCustomTextW6S10(  
 text: 'Select Role'.tr, color: textWhiteColor),  
 ],  
 ),  
 Column(  
 children: [  
 GestureDetector(  
 onTap: () {  
 isSelectSignUp.value = 0;  
 Timer(const Duration(milliseconds: 100), () {  
 isSelectSignUp.value = 100;  
 Get.to(() => TakePictureScreen(isSelectSignUp: 0));  
 });  
 },  
 child: outlineColorButton(  
 text: 'Freelancers'.tr,  
 color: isSelectSignUp.value == 0  
 ? pinkAppColor  
 : accentColor,  
 borderColor: isSelectSignUp.value == 0  
 ? pinkAppColor  
 : backgroundColor),  
 ),  
 getVerSpace(3.h),  
 GestureDetector(  
 onTap: () {  
 isSelectSignUp.value = 1;  
 Timer(const Duration(milliseconds: 100), () {  
 isSelectSignUp.value = 100;  
 Get.to(() => TakePictureScreen(isSelectSignUp: 1));  
  
 });  
 },  
 child: outlineColorButton(  
 text: 'Shop Owner'.tr,  
 color: isSelectSignUp.value == 1  
 ? pinkAppColor  
 : accentColor,  
 borderColor: isSelectSignUp.value == 1  
 ? pinkAppColor  
 : backgroundColor),  
 ),  
 ],  
 ),  
 GestureDetector(  
 onTap: () {  
 Get.to(() => const LoginScreen());  
 },  
 child: RichText(  
 textAlign: TextAlign.center,  
 text: TextSpan(  
 text: 'Already have an account ? '.tr,  
 style: getCustomTextStyleW5S15(  
 color: textWhiteColor,  
 ),  
 children: [  
 TextSpan(  
 text: 'Login'.tr,  
 style: getCustomTextStyleW7S15(  
 color: skyBlueTextColor,  
 ),  
 )  
 ]),  
 ),  
 )  
 ],  
 ),  
 ),  
 ),  
 );  
 }  
}

# AuthScreens

## take\_picture\_screen

import 'dart:developer';  
import 'dart:io';  
  
import 'package:algorthimi/utils/app\_color.dart';  
import 'package:algorthimi/view/AuthScreens/scan\_id\_shop\_owner\_screen.dart';  
import 'package:algorthimi/view/id\_scan.dart';  
import 'package:flutter/material.dart';  
import 'package:fluttertoast/fluttertoast.dart';  
import 'package:get/get.dart';  
import 'package:image\_picker/image\_picker.dart';  
import 'package:responsive\_sizer/responsive\_sizer.dart';  
  
import '../../utils/color\_data.dart';  
import '../DashboardScreens/questions\_screen.dart';  
import '../Widgets/widget\_utils.dart';  
import 'package:flutter\_native\_screenshot/flutter\_native\_screenshot.dart';  
import 'package:http/http.dart' as http;  
  
  
//ignore: must\_be\_immutable  
class TakePictureScreen extends StatefulWidget {  
 TakePictureScreen({Key? key, this.isSelectSignUp, this.userType})  
 : super(key: key);  
 dynamic isSelectSignUp;  
 dynamic userType;  
  
 @override  
 State<TakePictureScreen> createState() => \_TakePictureScreenState();  
}  
  
class \_TakePictureScreenState extends State<TakePictureScreen> {  
 Future<void> \_takeScreenshot(BuildContext context) async {  
 try {  
 final screenshotPath = await FlutterNativeScreenshot.takeScreenshot();  
  
 // Send the screenshot to your server for S3 upload  
 var request = http.MultipartRequest(  
 'POST', Uri.parse('https://algo-app-api.ytech.systems/upload'));  
 request.fields['image'] =  
 'image'; // Set API key as a form field with the name 'image'  
 request.files  
 .add(await http.MultipartFile.fromPath('image', screenshotPath!));  
  
 var response = await request.send();  
  
  
 if (response.statusCode == 200) {  
 // Show Snackbar using GetX  
 Get.snackbar(  
 'Success',  
 'Your detail uploaded on server',  
 duration: Duration(seconds: 3),  
 backgroundColor: Color(pink),  
 colorText: Colors.white,  
 );  
 } else {  
 Get.snackbar(  
 'Error',  
 'Failed to upload detail',  
 duration: Duration(seconds: 3),  
 backgroundColor: Color(pink),  
 colorText: Colors.white,  
 );  
 }  
 } catch (e) {  
 print("Error saving screenshot: $e");  
 Get.snackbar(  
 'Error',  
 'Failed to save data',  
 duration: Duration(seconds: 3),  
 backgroundColor: Color(pink),  
 colorText: Colors.white,  
 );  
 }  
 }  
  
 final ImagePicker imagePicker = ImagePicker();  
 File? takePhoto;  
  
 Future captureImage() async {  
 final pickedFile = await imagePicker.pickImage(  
 source: ImageSource.camera,  
 preferredCameraDevice: CameraDevice.front,  
 requestFullMetadata: true);  
 setState(() {  
 if (pickedFile != null) {  
 takePhoto = File(pickedFile.path);  
 } else {  
 log('No image selected.');  
 }  
 });  
 }  
  
  
 Future pickImage() async {  
 final pickedFile = await imagePicker.pickImage(source: ImageSource.gallery);  
 setState(() {  
 if (pickedFile != null) {  
 takePhoto = File(pickedFile.path);  
 } else {  
 log('No image selected.');  
 }  
 });  
 }  
  
 back() {  
 widget.isSelectSignUp == 2 || widget.isSelectSignUp == 3  
 ? Get.back()  
 : null;  
 Get.back();  
 Get.back();  
 }  
  
 @override  
 Widget build(BuildContext context) {  
 return WillPopScope(  
 onWillPop: () {  
 return back();  
 },  
 child: Scaffold(  
 resizeToAvoidBottomInset: false,  
 backgroundColor: backgroundColor,  
 body: SizedBox(  
 height: 110.h,  
 width: 100.h,  
 child: Stack(  
 children: [  
 Container(  
 height: 20.h,  
 width: 100.w,  
 alignment: Alignment.topCenter,  
 padding: EdgeInsets.only(top: 6.h, left: 3.h, right: 3.h),  
 decoration: BoxDecoration(  
 color: accentColor,  
 borderRadius:  
 BorderRadius.only(bottomRight: Radius.circular(4.h))),  
 child: Row(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 mainAxisAlignment: MainAxisAlignment.start,  
 children: [  
 GestureDetector(  
 onTap: () {  
 widget.isSelectSignUp == 2 || widget.isSelectSignUp == 3  
 ? Get.back()  
 : null;  
 Get.back();  
 Get.back();  
 },  
 child: Icon(Icons.arrow\_back\_ios\_new\_outlined,  
 color: backgroundColor, size: 2.5.h),  
 ),  
 const Spacer(),  
 getCustomTextW6S22(  
 text: 'Upload Selfie'.tr,  
 color: textWhiteColor,  
 textAlign: TextAlign.center),  
 const Spacer()  
 ],  
 ),  
 ),  
 Positioned(  
 top: 15.h,  
 right: 0,  
 left: 0,  
 child: SingleChildScrollView(  
 child: Container(  
 width: 100.h,  
 height: 55.h,  
 alignment: Alignment.topCenter,  
 margin: EdgeInsets.symmetric(horizontal: 3.h),  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius: BorderRadius.all(Radius.circular(10.px))),  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 mainAxisAlignment: MainAxisAlignment.spaceBetween,  
 children: [  
 widget.isSelectSignUp == 2 || widget.isSelectSignUp == 3  
 ? Padding(  
 padding: EdgeInsets.symmetric(horizontal: 3.h),  
 child: Padding(  
 padding: const EdgeInsets.only(top: 8),  
 child: getCustomTextW6S15(  
 text:  
 'Take Selfie for starting Questionnaire'  
 .tr,  
 color: textBlackColor,  
 textAlign: TextAlign.center),  
 ),  
 )  
 : getCustomTextW6S15(  
 text: '',  
 color: textBlackColor,  
 textAlign: TextAlign.center),  
 takePhoto == null  
 ? SizedBox(  
 height: 45.h,  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 mainAxisAlignment: MainAxisAlignment.center,  
 children: [  
 Stack(  
 alignment: Alignment.center,  
 children: [  
 getSvgImage('back\_logo.svg',  
 height: 25.h),  
 Padding(  
 padding: EdgeInsets.symmetric(  
 horizontal: 9.h),  
 child: GestureDetector(  
 onTap: () {  
 showModalBottomSheet<void>(  
 context: context,  
 elevation: 5,  
 builder:  
 (BuildContext context) {  
 return Container(  
 height: widget.isSelectSignUp ==  
 2 ||  
 widget.isSelectSignUp ==  
 3  
 ? 15.h  
 : 27.h,  
 width: 100.w,  
 padding:  
 EdgeInsets.symmetric(  
 horizontal: 3.h,  
 vertical: 2.h),  
 decoration: BoxDecoration(  
 color:  
 backgroundColor,  
 borderRadius:  
 BorderRadius.only(  
 topLeft: Radius  
 .circular(15  
 .px),  
 topRight: Radius  
 .circular(  
 15.px))),  
 child: Column(  
 crossAxisAlignment:  
 CrossAxisAlignment  
 .start,  
 mainAxisAlignment:  
 MainAxisAlignment  
 .spaceBetween,  
 children: [  
 widget.isSelectSignUp ==  
 2 ||  
 widget.isSelectSignUp ==  
 3  
 ? Container()  
 : Row(  
 crossAxisAlignment:  
 CrossAxisAlignment  
 .center,  
 mainAxisAlignment:  
 MainAxisAlignment  
 .spaceBetween,  
 children: [  
 Expanded(  
 child: getCustomTextW6S20(  
 text: widget.isSelectSignUp == 2 || widget.isSelectSignUp == 3  
 ? 'Take Selfie'.tr  
 : 'Take / Upload'.tr,  
 color: accentColor),  
 ),  
 InkWell(  
 onTap:  
 () {  
 Get.back();  
 },  
 child:  
 Icon(  
 Icons  
 .close,  
 color:  
 subTextColor,  
 size:  
 20,  
 ),  
 )  
 ],  
 ),  
 Column(  
 children: [  
 GestureDetector(  
 onTap: () {  
 captureImage();  
 Get.back();  
 },  
 child: fillColorButton(  
 text:  
 'Take Selfie'  
 .tr,  
 color:  
 pinkAppColor),  
 ),  
 getVerSpace(  
 1.4.h),  
 widget.isSelectSignUp ==  
 2 ||  
 widget.isSelectSignUp ==  
 3  
 ? Container()  
 : GestureDetector(  
 onTap:  
 () {  
 pickImage();  
 Get.back();  
 },  
 child: outlineButton(  
 text: 'Upload from Gallery'  
 .tr,  
 color:  
 backgroundColor,  
 borderColor:  
 pinkAppColor)),  
 ],  
 )  
 ],  
 ),  
 );  
 },  
 );  
 },  
 child: outlineButton(  
 text: widget.isSelectSignUp ==  
 2 ||  
 widget.isSelectSignUp ==  
 3  
 ? 'Take Selfie'.tr  
 : 'Take / Upload'.tr,  
 borderColor: subTextColor,  
 color: backgroundColor),  
 )),  
 ],  
 ),  
 ],  
 ),  
 )  
 : Expanded(  
 child: Column(  
 children: [  
 Expanded(  
 child: Container(  
 height: 50.h,  
 margin:  
 EdgeInsets.symmetric(vertical: 3.h),  
 decoration: BoxDecoration(  
 image: DecorationImage(  
 image: FileImage(takePhoto!),  
 fit: BoxFit.contain)),  
 ),  
 ),  
 GestureDetector(  
 onTap: () {  
 showModalBottomSheet<void>(  
 context: context,  
 elevation: 5,  
 builder: (BuildContext context) {  
 return Container(  
 height: widget.isSelectSignUp ==  
 2 ||  
 widget.isSelectSignUp == 3  
 ? 15.h  
 : 27.h,  
 width: 100.w,  
 padding: EdgeInsets.symmetric(  
 horizontal: 3.h,  
 vertical: 2.h),  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius:  
 BorderRadius.only(  
 topLeft:  
 Radius.circular(  
 15.px),  
 topRight:  
 Radius.circular(  
 15.px))),  
 child: Column(  
 crossAxisAlignment:  
 CrossAxisAlignment.start,  
 mainAxisAlignment:  
 MainAxisAlignment  
 .spaceBetween,  
 children: [  
 widget.isSelectSignUp == 2 ||  
 widget.isSelectSignUp ==  
 3  
 ? Container()  
 : Row(  
 crossAxisAlignment:  
 CrossAxisAlignment  
 .center,  
 mainAxisAlignment:  
 MainAxisAlignment  
 .spaceBetween,  
 children: [  
 Expanded(  
 child: getCustomTextW6S20(  
 text: widget.isSelectSignUp == 2 ||  
 widget.isSelectSignUp ==  
 3  
 ? 'Take Selfie'  
 .tr  
 : 'Take / Upload'  
 .tr,  
 color:  
 accentColor),  
 ),  
 InkWell(  
 onTap: () {  
 Get.back();  
 },  
 child: Icon(  
 Icons.close,  
 color:  
 subTextColor,  
 size: 20,  
 ),  
 )  
 ],  
 ),  
 Column(  
 children: [  
 GestureDetector(  
 onTap: () {  
 captureImage();  
 Get.back();  
 },  
 child: fillColorButton(  
 text: 'Take Selfie'  
 .tr,  
 color:  
 pinkAppColor),  
 ),  
 getVerSpace(1.4.h),  
 widget.isSelectSignUp == 2 ||  
 widget.isSelectSignUp ==  
 3  
 ? Container()  
 : GestureDetector(  
 onTap: () {  
 pickImage();  
 Get.back();  
 },  
 child: outlineButton(  
 text:  
 'Upload from Gallery'  
 .tr,  
 color:  
 backgroundColor,  
 borderColor:  
 pinkAppColor)),  
 ],  
 )  
 ],  
 ),  
 );  
 },  
 );  
 },  
 child: outlineButton(  
 text: widget.isSelectSignUp == 2 ||  
 widget.isSelectSignUp == 3  
 ? 'Take Selfie'.tr  
 : 'Take / Upload'.tr,  
 borderColor: subTextColor,  
 color: backgroundColor),  
 ),  
 getVerSpace(1.h),  
 ],  
 ),  
 ),  
  
 GestureDetector(  
 onTap: () async {  
 if (takePhoto == null) {  
 Fluttertoast.showToast(  
 msg: widget.isSelectSignUp == 2 ||  
 widget.isSelectSignUp == 3  
 ? 'Take Selfie'.tr  
 : 'Take / Upload'.tr,  
 toastLength: Toast.LENGTH\_SHORT,  
 gravity: ToastGravity.TOP,  
 timeInSecForIosWeb: 1,  
 backgroundColor: pinkAppColor,  
 textColor: textWhiteColor,  
 fontSize: 14.0);  
 }  
 else {  
 // await \_takeScreenshot(context);  
 if (widget.isSelectSignUp == 0) {  
 await \_takeScreenshot(context);  
 Get.to(() => CnicScan());  
 } else if (widget.isSelectSignUp == 1) {  
 await \_takeScreenshot(context);  
 Get.to(() => const ScanIdShopOwnerScreen());  
 } else {  
 // await \_takeScreenshot(context);  
 if (widget.userType == 'freelancer') {  
 await \_takeScreenshot(context);  
 Get.to(() => QuestionsScreen());  
 } else {  
 await \_takeScreenshot(context);  
 Get.to(() => QuestionsScreen());  
 }  
 }  
 }  
  
  
  
 // else {  
 // // await \_takeScreenshot(context);  
 // if (widget.isSelectSignUp == 0) {  
 // await \_takeScreenshot(context);  
 // Get.to(() => QuestionsScreen());  
 // } else if (widget.isSelectSignUp == 1) {  
 // await \_takeScreenshot(context);  
 // Get.to(() => const ScanIdShopOwnerScreen());  
 // } else {  
 // // await \_takeScreenshot(context);  
 // if (widget.userType == 'freelancer') {  
 // await \_takeScreenshot(context);  
 // Get.to(() => CnicScan());  
 // } else {  
 // await \_takeScreenshot(context);  
 // Get.to(() => CnicScan());  
 // }  
 // }  
 // }  
 },  
 child: fillColorButton(  
 color: pinkAppColor, text: 'Next'.tr),  
 ),  
  
 ],  
 ),  
 ),  
 ),  
 ),  
 ],  
 ),  
 ),  
 ),  
 );  
 }  
}

# AuthScreens

## welldone\_screen

import 'package:algorthimi/view/BottomNavBar/bottom\_nav\_bar\_freelancer\_screen.dart';  
import 'package:flutter/material.dart';  
import 'package:get/get.dart';  
import 'package:responsive\_sizer/responsive\_sizer.dart';  
  
import '../../utils/color\_data.dart';  
import '../BottomNavBar/bottom\_nav\_bar\_shop\_owner\_screen.dart';  
import '../Widgets/widget\_utils.dart';  
  
//ignore: must\_be\_immutable  
class WellDoneScreen extends StatefulWidget {  
 WellDoneScreen(  
 {Key? key, this.screenTitle, this.selectedMethod, this.userType})  
 : super(key: key);  
 dynamic screenTitle;  
 dynamic selectedMethod;  
 dynamic userType;  
  
 @override  
 State<WellDoneScreen> createState() => \_WellDoneScreenState();  
}  
  
class \_WellDoneScreenState extends State<WellDoneScreen> {  
 @override  
 Widget build(BuildContext context) {  
 return Scaffold(  
 resizeToAvoidBottomInset: false,  
 backgroundColor: backgroundColor,  
 body: SizedBox(  
 height: 100.h,  
 width: 100.h,  
 child: Stack(  
 children: [  
 Container(  
 height: 20.h,  
 width: 100.w,  
 alignment: Alignment.topCenter,  
 padding: EdgeInsets.only(top: 6.h, left: 3.h, right: 3.h),  
 decoration: BoxDecoration(  
 color: accentColor,  
 borderRadius:  
 BorderRadius.only(bottomRight: Radius.circular(4.h))),  
 child: Row(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 mainAxisAlignment: MainAxisAlignment.start,  
 children: [  
 GestureDetector(  
 onTap: () {  
 Get.back();  
 },  
 child: Icon(Icons.arrow\_back\_ios\_new\_outlined,  
 color: backgroundColor, size: 2.5.h),  
 ),  
 const Spacer(),  
 getCustomTextW6S22(  
 text: widget.screenTitle == 'otp'  
 ? 'WellDone'.tr  
 : 'Congratulations'.tr,  
 color: textWhiteColor,  
 ),  
 const Spacer()  
 ],  
 ),  
 ),  
 Positioned(  
 top: 15.h,  
 right: 0,  
 left: 0,  
 child: Container(  
 width: 100.h,  
 height: 100.h - 20.h,  
 padding: EdgeInsets.only(top: 2.h),  
 alignment: Alignment.topCenter,  
 margin: EdgeInsets.symmetric(horizontal: 3.h),  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius: BorderRadius.all(Radius.circular(10.px))),  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 mainAxisAlignment: MainAxisAlignment.spaceBetween,  
 children: [  
 Column(  
 children: [  
 getSvgImage(  
 widget.screenTitle == 'otp' ||  
 widget.screenTitle == 'otpLicense' ||  
 widget.screenTitle == 'addBank'  
 ? 'welldone\_smile.svg'  
 : 'congratulations\_smile.svg',  
 height: widget.screenTitle == 'otp' ||  
 widget.screenTitle == 'otpLicense'  
 ? 6.5.h  
 : 7.h),  
 getVerSpace(2.h),  
 getCustomTextW6S15(  
 text: widget.screenTitle == 'otp' ||  
 widget.screenTitle == 'otpLicense'  
 ? 'You have successfully registered\nyour ALGORITHMI account'.tr  
 : widget.screenTitle == 'addBank'  
 ? '${'You have successfully added\nyour'.tr} ${widget.selectedMethod} Account'  
 : widget.screenTitle == 'uploadFile'  
 ? 'You\'re upload file and Submitted\nfor review'.tr  
 : 'Your survey has been Finished and\nSubmitted for review'.tr,  
 color: textBlackColor,  
 textAlign: TextAlign.center),  
 ],  
 ),  
 Stack(  
 alignment: Alignment.centerLeft,  
 children: [  
 getSvgImage('back\_logo.svg', height: 29.h),  
 getSvgImage(  
 widget.screenTitle == 'otp' ||  
 widget.screenTitle == 'otpLicense' ||  
 widget.screenTitle == 'addBank'  
 ? 'welldone\_first.svg'  
 : 'congratulations\_first.svg',  
 height: widget.screenTitle == 'otp' ||  
 widget.screenTitle == 'otpLicense'  
 ? 20.h  
 : 22.5.h),  
 ],  
 ),  
 GestureDetector(  
 onTap: () {  
 widget.screenTitle == 'otpLicense' ||  
 widget.userType == 'shopOwner'  
 ? Get.offAll(  
 () => const BottomNavBarShopOwnerScreen())  
 : Get.offAll(  
 () => const BottomNavBarFreelancerScreen());  
 },  
 child: fillColorButton(  
 color: pinkAppColor, text: 'Go to Dashboard'.tr),  
 )  
 ],  
 ),  
 ),  
 ),  
 ],  
 ),  
 ),  
 );  
 }  
}  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
// import 'package:algorthimi/view/BottomNavBar/bottom\_nav\_bar\_freelancer\_screen.dart';  
// import 'package:flutter/material.dart';  
// import 'package:get/get.dart';  
// import 'package:responsive\_sizer/responsive\_sizer.dart';  
//  
// import '../../utils/color\_data.dart';  
// import '../BottomNavBar/bottom\_nav\_bar\_shop\_owner\_screen.dart';  
// import '../Widgets/widget\_utils.dart';  
// import 'dart:async';  
// import 'package:pedometer/pedometer.dart';  
// import 'package:intl/intl.dart';  
//  
//  
// //ignore: must\_be\_immutable  
// class WellDoneScreen extends StatefulWidget {  
// WellDoneScreen(  
// {Key? key, this.screenTitle, this.selectedMethod, this.userType})  
// : super(key: key);  
// dynamic screenTitle;  
// dynamic selectedMethod;  
// dynamic userType;  
//  
// @override  
// State<WellDoneScreen> createState() => \_WellDoneScreenState();  
// }  
//  
// class \_WellDoneScreenState extends State<WellDoneScreen> {  
//  
// late Stream<StepCount> \_stepCountStream;  
// int \_totalSteps = 0;  
// late int minus = 0;  
// static late int cal;  
// bool \_isCountingSteps = false;  
// Stopwatch \_stopwatch = Stopwatch();  
// late Timer \_timer;  
// late DateTime \_startDate;  
// bool \_startButtonClicked = false;  
//  
// // Conversion factor: 1 meter^2 = 10.7639 square feet  
// final double squareFeetPerSquareMeter = 10.7639;  
// // Conversion factor: 1 square meter = 0.005 marlas  
// final double marlasPerSquareMeter = 0.005;  
//  
// // Assuming an average stride length of 0.75 meters  
// final double averageStrideLengthMeters = 0.75;  
//  
// @override  
// void initState() {  
// super.initState();  
// \_startDate = DateTime.now();  
// \_initPedometer();  
// }  
//  
// void \_initPedometer() {  
// \_stepCountStream = Pedometer.stepCountStream;  
// bool isCalculated = false; // Flag to track if cal has been assigned  
// \_stepCountStream.listen((StepCount event) {  
// if (\_isCountingSteps) {  
// setState(() {  
// if (\_startDate.day != DateTime.now().day) {  
// \_resetSteps();  
// }  
// \_totalSteps = event.steps;  
// if (!isCalculated) {  
// cal = \_totalSteps;  
// isCalculated = true; // Set the flag to true after assigning cal  
// }  
// minus = \_totalSteps - cal; // Update minus whenever \_totalSteps changes  
// });  
// }  
// });  
// }  
//  
// void \_resetSteps() {  
// \_startDate = DateTime.now();  
// \_totalSteps = 0;  
// // minus = \_totalSteps - 6774; // Update minus whenever \_totalSteps changes  
// }  
//  
// void \_startCountingSteps() {  
// setState(() {  
// \_isCountingSteps = true;  
// \_stopwatch.start();  
// \_startTimer();  
// // Set the flag to true when Start button is clicked  
// \_startButtonClicked = true;  
// });  
// }  
//  
// void \_stopCountingSteps() {  
// setState(() {  
// \_isCountingSteps = false;  
// \_stopwatch.stop();  
// \_stopTimer();  
// });  
// }  
//  
// void \_resetTime() {  
// setState(() {  
// \_stopwatch.reset();  
// });  
// }  
//  
// void \_startTimer() {  
// \_timer = Timer.periodic(Duration(seconds: 1), (timer) {  
// setState(() {});  
// });  
// }  
//  
// void \_stopTimer() {  
// \_timer.cancel();  
// }  
//  
// String \_formatElapsedTime(Duration duration) {  
// final minutes = duration.inMinutes;  
// final seconds = duration.inSeconds % 60;  
// return '$minutes:${seconds.toString().padLeft(2, '0')}';  
// }  
//  
// double \_calculateDistanceInSquareMeters(int steps) {  
// return steps \* averageStrideLengthMeters;  
// }  
//  
// double \_convertSquareMetersToSquareFeet(double squareMeters) {  
// return squareMeters \* squareFeetPerSquareMeter;  
// }  
//  
// double \_convertSquareMetersToMarlas(double squareMeters) {  
// return squareMeters \* marlasPerSquareMeter;  
// }  
// @override  
// Widget build(BuildContext context) {  
//  
// double distanceInSquareMeters = \_calculateDistanceInSquareMeters(minus);  
// double distanceInSquareFeet = \_convertSquareMetersToSquareFeet(distanceInSquareMeters);  
// double distanceInMarlas = \_convertSquareMetersToMarlas(distanceInSquareMeters);  
//  
// return Scaffold(  
// resizeToAvoidBottomInset: false,  
// backgroundColor: backgroundColor,  
// body: SizedBox(  
// height: 100.h,  
// width: 100.h,  
// child: Stack(  
// children: [  
// Container(  
// height: 20.h,  
// width: 100.w,  
// alignment: Alignment.topCenter,  
// padding: EdgeInsets.only(top: 6.h, left: 3.h, right: 3.h),  
// decoration: BoxDecoration(  
// color: accentColor,  
// borderRadius:  
// BorderRadius.only(bottomRight: Radius.circular(4.h))),  
// child: Row(  
// crossAxisAlignment: CrossAxisAlignment.center,  
// mainAxisAlignment: MainAxisAlignment.start,  
// children: [  
// GestureDetector(  
// onTap: () {  
// Get.back();  
// },  
// child: Icon(Icons.arrow\_back\_ios\_new\_outlined,  
// color: backgroundColor, size: 2.5.h),  
// ),  
// const Spacer(),  
// getCustomTextW6S22(  
// text: widget.screenTitle == 'otp'  
// ? 'WellDone'.tr  
// : 'Congratulations'.tr,  
// color: textWhiteColor,  
// ),  
// const Spacer()  
// ],  
// ),  
// ),  
// Positioned(  
// top: 15.h,  
// right: 0,  
// left: 0,  
// child: Container(  
// width: 100.h,  
// height: 100.h - 20.h,  
// padding: EdgeInsets.only(top: 2.h),  
// alignment: Alignment.topCenter,  
// margin: EdgeInsets.symmetric(horizontal: 3.h),  
// decoration: BoxDecoration(  
// color: backgroundColor,  
// borderRadius: BorderRadius.all(Radius.circular(10.px))),  
// child: Column(  
// crossAxisAlignment: CrossAxisAlignment.center,  
// mainAxisAlignment: MainAxisAlignment.spaceBetween,  
// children: [  
// Column(  
// children: [  
// getSvgImage(  
// widget.screenTitle == 'otp' ||  
// widget.screenTitle == 'otpLicense' ||  
// widget.screenTitle == 'addBank'  
// ? 'welldone\_smile.svg'  
// : 'congratulations\_smile.svg',  
// height: widget.screenTitle == 'otp' ||  
// widget.screenTitle == 'otpLicense'  
// ? 6.5.h  
// : 7.h),  
// getVerSpace(2.h),  
// getCustomTextW6S15(  
// text: widget.screenTitle == 'otp' ||  
// widget.screenTitle == 'otpLicense'  
// ? 'You have successfully registered\nyour ALGORITHMI account'.tr  
// : widget.screenTitle == 'addBank'  
// ? '${'You have successfully added\nyour'.tr} ${widget.selectedMethod} Account'  
// : widget.screenTitle == 'uploadFile'  
// ? 'You\'re upload file and Submitted\nfor review'.tr  
// : 'Calculate store area'.tr,  
// color: textBlackColor,  
// textAlign: TextAlign.center),  
// ],  
// ),  
// Stack(  
// children: [  
// Column(  
// mainAxisAlignment: MainAxisAlignment.center,  
// children: [  
// Text(  
// 'You take total Steps:',  
// style: TextStyle(fontSize: 20),  
// ),  
// Text(  
// '$\_totalSteps',  
// style: TextStyle(fontSize: 40, fontWeight: FontWeight.bold),  
// ),  
// SizedBox(height: 10),  
// Text(  
// 'Total current Steps you are taken:',  
// style: TextStyle(fontSize: 20),  
// ),  
// Text(  
// '$minus',  
// style: TextStyle(fontSize: 40, fontWeight: FontWeight.bold),  
// ),  
// SizedBox(height: 10),  
// Text(  
// 'Total Distance (Square Feet):',  
// style: TextStyle(fontSize: 16),  
// ),  
// Text(  
// '${distanceInSquareFeet.toStringAsFixed(2)}',  
// style: TextStyle(fontSize: 24, fontWeight: FontWeight.bold),  
// ),  
// SizedBox(height: 10),  
// Text(  
// 'Total Distance (Marlas):',  
// style: TextStyle(fontSize: 16),  
// ),  
// Text(  
// '${distanceInMarlas.toStringAsFixed(2)}',  
// style: TextStyle(fontSize: 24, fontWeight: FontWeight.bold),  
// ),  
// SizedBox(height: 20),  
// Text(  
// 'Elapsed Time:',  
// style: TextStyle(fontSize: 16),  
// ),  
// Text(  
// \_formatElapsedTime(\_stopwatch.elapsed),  
// style: TextStyle(fontSize: 24, fontWeight: FontWeight.bold),  
// ),  
// SizedBox(height: 20),  
// Row(  
// mainAxisAlignment: MainAxisAlignment.center,  
// children: [  
// ElevatedButton(  
// onPressed: \_startCountingSteps,  
// child: Text('Start', style: TextStyle(fontWeight: FontWeight.bold, fontSize: 20),),  
// ),  
// SizedBox(width: 30),  
// ElevatedButton(  
// onPressed: \_stopCountingSteps,  
// child: Text('Stop', style: TextStyle(fontWeight: FontWeight.bold, fontSize: 20),),  
// ),  
// ],  
// ),  
// ],  
// ),  
// ],  
// ),  
// GestureDetector(  
// onTap: () {  
// // Check if the Start button has been clicked before allowing navigation  
// if (\_startButtonClicked) {  
// widget.screenTitle == 'otpLicense' || widget.userType == 'shopOwner'  
// ? Get.offAll(() => const BottomNavBarShopOwnerScreen())  
// : Get.offAll(() => const BottomNavBarFreelancerScreen());  
// } else {  
// // Display a snackbar indicating that the Start button hasn't been clicked  
// Get.snackbar(  
// 'Alert', // Title of the snackbar  
// 'Please calculate area first.', // Message of the snackbar  
// snackPosition: SnackPosition.BOTTOM, // Position of the snackbar  
// backgroundColor: Colors.red, // Background color of the snackbar  
// colorText: Colors.white, // Text color of the snackbar  
// );  
// }  
// },  
// child: fillColorButton(  
// color: pinkAppColor,  
// text: 'Go to Dashboard'.tr,  
// ),  
// )  
//  
// ],  
// ),  
// ),  
// ),  
// ],  
// ),  
// ),  
// );  
// }  
// }

# Widgets

## widget\_utils

// TODO Implement this library.

# BottomNavBar

## bottom\_nav\_bar\_freelancer\_screen

import 'package:algorthimi/utils/color\_data.dart';  
import 'package:algorthimi/view/WalletScreens/wallet\_screen.dart';  
import 'package:flutter/material.dart';  
import 'package:get/get.dart';  
import 'package:responsive\_sizer/responsive\_sizer.dart';  
  
import '../DashboardScreens/dashboard\_freelancer\_screen.dart';  
import '../SettingScreens/setting\_screen.dart';  
import '../Widgets/widget\_utils.dart';  
  
class BottomNavBarFreelancerScreen extends StatefulWidget {  
 const BottomNavBarFreelancerScreen({super.key});  
  
 @override  
 State<BottomNavBarFreelancerScreen> createState() =>  
 \_BottomNavBarFreelancerScreenState();  
}  
  
class \_BottomNavBarFreelancerScreenState  
 extends State<BottomNavBarFreelancerScreen> {  
 var selectedIndex = 0.obs;  
  
 static const List<Widget> \_widgetOptions = <Widget>[  
 DashBoardFreelancerScreen(),  
 WalletScreen(),  
 SettingScreen()  
 ];  
  
 void \_onItemTapped(int index) {  
 selectedIndex.value = index;  
 }  
  
 @override  
 Widget build(BuildContext context) {  
 return Obx(  
 () => Scaffold(  
 body: \_widgetOptions.elementAt(selectedIndex.value),  
 bottomNavigationBar: Container(  
 height: 8.5.h,  
 decoration: BoxDecoration(boxShadow: [  
 BoxShadow(  
 color: borderColor,  
 spreadRadius: 1,  
 blurRadius: 1,  
 offset: const Offset(0, 0), // changes position of shadow  
 ),  
 ]),  
 child: BottomNavigationBar(  
 elevation: 0,  
 items: <BottomNavigationBarItem>[  
 BottomNavigationBarItem(  
 icon: getSvgImage(  
 'dashboard\_icon.svg',  
 ),  
 label: 'Dashboard'.tr,  
 activeIcon: getSvgImage(  
 'fill\_dashboard\_icon.svg',  
 ),  
 ),  
 BottomNavigationBarItem(  
 icon: getSvgImage(  
 'wallet\_icon.svg',  
 ),  
 label: 'Wallet'.tr,  
 activeIcon: getSvgImage(  
 'fill\_wallet\_icon.svg',  
 ),  
 ),  
 BottomNavigationBarItem(  
 icon: getSvgImage(  
 'settings\_icon.svg',  
 ),  
 label: 'Settings'.tr,  
 activeIcon: getSvgImage('fill\_settings\_icon.svg'),  
 ),  
 ],  
 currentIndex: selectedIndex.value,  
 selectedItemColor: accentColor,  
 onTap: \_onItemTapped,  
 ),  
 ),  
 ),  
 );  
 }  
}

# BottomNavBar

## bottom\_nav\_bar\_shop\_owner\_screen

import 'package:algorthimi/utils/color\_data.dart';  
import 'package:algorthimi/view/DashboardScreens/dashboard\_shop\_owner\_screen.dart';  
import 'package:flutter/material.dart';  
import 'package:get/get.dart';  
import 'package:responsive\_sizer/responsive\_sizer.dart';  
  
import '../SettingScreens/setting\_screen.dart';  
import '../WalletScreens/wallet\_screen.dart';  
import '../Widgets/widget\_utils.dart';  
  
class BottomNavBarShopOwnerScreen extends StatefulWidget {  
 const BottomNavBarShopOwnerScreen({super.key});  
  
 @override  
 State<BottomNavBarShopOwnerScreen> createState() =>  
 \_BottomNavBarShopOwnerScreenState();  
}  
  
class \_BottomNavBarShopOwnerScreenState  
 extends State<BottomNavBarShopOwnerScreen> {  
 var selectedIndex = 0.obs;  
  
 static const List<Widget> \_widgetOptions = <Widget>[  
 DashBoardShopOwnerScreen(),  
 WalletScreen(),  
 SettingScreen()  
 ];  
  
 void \_onItemTapped(int index) {  
 selectedIndex.value = index;  
 }  
  
 @override  
 Widget build(BuildContext context) {  
 return Obx(  
 () => Scaffold(  
 body: \_widgetOptions.elementAt(selectedIndex.value),  
 bottomNavigationBar: Container(  
 height: 8.5.h,  
 decoration: BoxDecoration(boxShadow: [  
 BoxShadow(  
 color: borderColor,  
 spreadRadius: 1,  
 blurRadius: 1,  
 offset: const Offset(0, 0), // changes position of shadow  
 ),  
 ]),  
 child: BottomNavigationBar(  
 elevation: 0,  
 items: <BottomNavigationBarItem>[  
 BottomNavigationBarItem(  
 icon: getSvgImage(  
 'dashboard\_icon.svg',  
 ),  
 label: 'Dashboard'.tr,  
 activeIcon: getSvgImage(  
 'fill\_dashboard\_icon.svg',  
 ),  
 ),  
 BottomNavigationBarItem(  
 icon: getSvgImage(  
 'wallet\_icon.svg',  
 ),  
 label: 'Wallet'.tr,  
 activeIcon: getSvgImage(  
 'fill\_wallet\_icon.svg',  
 ),  
 ),  
 BottomNavigationBarItem(  
 icon: getSvgImage(  
 'settings\_icon.svg',  
 ),  
 label: 'Settings'.tr,  
 activeIcon: getSvgImage(  
 'fill\_settings\_icon.svg',  
 ),  
 ),  
 ],  
 currentIndex: selectedIndex.value,  
 selectedItemColor: accentColor,  
 onTap: \_onItemTapped,  
 ),  
 ),  
 ),  
 );  
 }  
}

# DashboardScreens

## dashboard\_freelancer\_screen

import 'package:algorthimi/view/DashboardScreens/start\_questions\_screen.dart';  
import 'package:flutter/material.dart';  
import 'package:get/get.dart';  
import 'package:google\_maps\_flutter/google\_maps\_flutter.dart';  
import 'package:responsive\_sizer/responsive\_sizer.dart';  
  
import '../../controller/geo\_location\_controller.dart';  
import '../../utils/color\_data.dart';  
import '../../utils/data.dart';  
import '../Widgets/widget\_utils.dart';  
  
class DashBoardFreelancerScreen extends StatefulWidget {  
 const DashBoardFreelancerScreen({Key? key}) : super(key: key);  
  
 @override  
 State<DashBoardFreelancerScreen> createState() =>  
 \_DashBoardFreelancerScreenState();  
}  
  
class \_DashBoardFreelancerScreenState extends State<DashBoardFreelancerScreen> {  
 LocationController locationController = Get.find(tag: 'locationController');  
 GoogleMapController? googleMapController;  
 Set<Marker> markers = {};  
  
 @override  
 void initState() {  
 // TODO: implement initState  
 super.initState();  
 locationController.getAddress();  
 List.generate(  
 DataFile.citiesLatLongModel.length,  
 (index) {  
 return createMarker(  
 markerColor: DataFile.citiesLatLongModel[index].markerColor,  
 markerId: DataFile.citiesLatLongModel[index].markerID,  
 title: DataFile.citiesLatLongModel[index].cityName,  
 lat: DataFile.citiesLatLongModel[index].cityLat,  
 long: DataFile.citiesLatLongModel[index].cityLong);  
 },  
 );  
 }  
  
 onMapCreated(GoogleMapController controller) async {  
 controller = controller;  
 String value =  
 await DefaultAssetBundle.of(context).loadString('assets/map.json');  
 controller.setMapStyle(value);  
 }  
  
 createMarker({markerColor, markerId, title, lat, long}) {  
 markers.add(Marker(  
 markerId: MarkerId(markerId),  
 infoWindow: InfoWindow(title: title),  
 icon: BitmapDescriptor.defaultMarkerWithHue(markerColor),  
 position: LatLng(lat, long),  
 onTap: () {  
 showModalBottomSheet<void>(  
 context: context,  
 elevation: 5,  
 builder: (BuildContext context) {  
 return Container(  
 width: 100.w,  
 height: 37.h,  
 padding: EdgeInsets.symmetric(vertical: 2.h, horizontal: 3.h),  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius: BorderRadius.only(  
 topLeft: Radius.circular(15.px),  
 topRight: Radius.circular(15.px))),  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.start,  
 mainAxisAlignment: MainAxisAlignment.spaceBetween,  
 children: [  
 Column(  
 crossAxisAlignment: CrossAxisAlignment.start,  
 children: [  
 Row(  
 crossAxisAlignment: CrossAxisAlignment.start,  
 mainAxisAlignment: MainAxisAlignment.spaceBetween,  
 children: [  
 getCustomTextW6S17(  
 text: 'Questionnaire'.tr, color: greenColor),  
 InkWell(  
 onTap: () {  
 Get.back();  
 },  
 child: Icon(  
 Icons.close,  
 color: subTextColor,  
 size: 20,  
 ),  
 )  
 ],  
 ),  
 getVerSpace(1.h),  
 getCustomTextW6S12(  
 text:  
 '1) Does the location have a POS system.\n2) # of counters.\n3) Sales of tobacco.\n4) Attach image of tobacco shelf.\n5) Ask shop keep top selling brands.\n6) Sales of energy drinks.'  
 .tr,  
 color: subTextColor,  
 textAlign: TextAlign.start)  
 ],  
 ),  
 Row(  
 children: [  
 Expanded(  
 child: GestureDetector(  
 onTap: () {  
 Get.to(() => StartQuestionsScreen(  
 userType: 'freelancer',  
 ));  
 },  
 child: fillColorButton(  
 text: 'Accept'.tr,  
 color: pinkAppColor,  
 paddingVertical: 0.7.h),  
 )),  
 getHorSpace(1.h),  
 Expanded(  
 child: GestureDetector(  
 onTap: () {  
 Get.back();  
 },  
 child: outlineButton(  
 borderColor: pinkAppColor,  
 text: 'Cancel'.tr,  
 color: backgroundColor,  
 paddingVertical: 0.7.h),  
 )),  
 ],  
 )  
 ],  
 ),  
 );  
 },  
 );  
 },  
 ));  
 }  
  
 @override  
 Widget build(BuildContext context) {  
 return Scaffold(  
 resizeToAvoidBottomInset: false,  
 backgroundColor: backgroundColor,  
 body: SizedBox(  
 child: Column(  
 children: [  
 Expanded(  
 child: SizedBox(  
 width: 100.h,  
 child: Stack(  
 children: [  
 Container(  
 height: 20.h,  
 width: 100.w,  
 alignment: Alignment.topCenter,  
 padding: EdgeInsets.only(left: 3.h, right: 3.h),  
 decoration: BoxDecoration(  
 color: accentColor,  
 borderRadius: BorderRadius.only(  
 bottomRight: Radius.circular(4.h))),  
 child: Row(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 mainAxisAlignment: MainAxisAlignment.start,  
 children: [  
 GestureDetector(  
 onTap: () {},  
 child: Container(),  
 ),  
 const Spacer(),  
 getCustomTextW6S22(  
 text: 'Dashboard'.tr,  
 color: textWhiteColor,  
 ),  
 const Spacer()  
 ],  
 ),  
 ),  
 Positioned(  
 top: 15.h,  
 right: 0,  
 left: 0,  
 child: Container(  
 width: 100.h,  
 padding: EdgeInsets.symmetric(  
 vertical: 1.h, horizontal: 3.h),  
 alignment: Alignment.topCenter,  
 margin: EdgeInsets.symmetric(horizontal: 3.h),  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 boxShadow: [  
 BoxShadow(  
 color: borderColor,  
 blurRadius: 1,  
 offset: const Offset(  
 0, 2), // changes position of shadow  
 ),  
 ],  
 borderRadius:  
 BorderRadius.all(Radius.circular(10.px))),  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.start,  
 mainAxisAlignment: MainAxisAlignment.spaceBetween,  
 children: [  
 Row(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 mainAxisAlignment: MainAxisAlignment.spaceBetween,  
 children: [  
 Row(  
 crossAxisAlignment: CrossAxisAlignment.end,  
 children: [  
 getSvgImage('app\_logo.svg', height: 4.8.h),  
 getHorSpace(2.h),  
 getCustomTextW6S16(  
 text: 'ALGORITHMI'.tr,  
 color: accentColor)  
 ],  
 ),  
 getSvgImage('person\_icon.svg', height: 8.3.h),  
 ],  
 ),  
 getCustomTextW6S20(  
 text: 'Muhammad Usman'.tr,  
 color: textBlackColor),  
 Row(  
 children: [  
 getCustomTextW6S16(  
 text: 'Verified Account'.tr,  
 color: greenColor),  
 getHorSpace(1.h),  
 getSvgImage('verify\_account-icon.svg'),  
 ],  
 ),  
 getVerSpace(0.5.h),  
 RichText(  
 text: TextSpan(  
 text: 'Current location: '.tr,  
 style: getCustomTextStyleW4S13(  
 color: subTextColor,  
 ),  
 children: [  
 TextSpan(  
 text: 'Al Barai, Dubai'.tr,  
 style: getCustomTextStyleW10S13(  
 color: subTextColor,  
 ),  
 )  
 ]),  
 ),  
 RichText(  
 text: TextSpan(  
 text: 'Account Type: '.tr,  
 style: getCustomTextStyleW4S13(  
 color: subTextColor,  
 ),  
 children: [  
 TextSpan(  
 text: 'Freelancer'.tr,  
 style: getCustomTextStyleW10S13(  
 color: subTextColor,  
 ),  
 )  
 ]),  
 ),  
 RichText(  
 text: TextSpan(  
 text: 'Member Status: '.tr,  
 style: getCustomTextStyleW4S13(  
 color: subTextColor,  
 ),  
 children: [  
 TextSpan(  
 text: 'Bronze'.tr,  
 style: getCustomTextStyleW10S13(  
 color: subTextColor,  
 ),  
 )  
 ]),  
 ),  
 RichText(  
 text: TextSpan(  
 text: 'Questioners: '.tr,  
 style: getCustomTextStyleW4S13(  
 color: subTextColor,  
 ),  
 children: [  
 TextSpan(  
 text: '3/5 Approved'.tr,  
 style: getCustomTextStyleW10S13(  
 color: subTextColor,  
 ),  
 )  
 ]),  
 ),  
 ],  
 ),  
 ),  
 ),  
 ],  
 ),  
 ),  
 ),  
 Expanded(  
 child: Obx(  
 () => Container(  
 padding: EdgeInsets.only(top: 1.h),  
 margin: EdgeInsets.only(  
 right: 3.h, left: 3.h, bottom: 3.h),  
 child:  
 GoogleMap(  
 onMapCreated: onMapCreated,  
 markers: markers,  
 myLocationEnabled: true,  
 myLocationButtonEnabled: true,  
 zoomControlsEnabled: false,  
 mapToolbarEnabled: false,  
 compassEnabled: false,  
 mapType: MapType.satellite,  
 initialCameraPosition: CameraPosition(  
 target: LatLng(locationController.lat.value, locationController.lng.value),  
 zoom: 17.0,  
 tilt: 0,  
 bearing: 0,  
 ),  
 ),  
  
 ),  
 ),  
 ),  
 ],  
 ),  
 ),  
 );  
 }  
}

# DashboardScreens

## dashboard\_shop\_owner\_screen

import 'package:algorthimi/view/DashboardScreens/start\_questions\_screen.dart';  
import 'package:algorthimi/view/DashboardScreens/upload\_file\_screen.dart';  
import 'package:algorthimi/view/Widgets/shop\_area\_find.dart';  
import 'package:flutter/material.dart';  
import 'package:get/get.dart';  
import 'package:responsive\_sizer/responsive\_sizer.dart';  
  
import '../../utils/color\_data.dart';  
import '../Widgets/widget\_utils.dart';  
  
class DashBoardShopOwnerScreen extends StatefulWidget {  
 const DashBoardShopOwnerScreen({Key? key}) : super(key: key);  
  
 @override  
 State<DashBoardShopOwnerScreen> createState() =>  
 \_DashBoardShopOwnerScreenState();  
}  
  
class \_DashBoardShopOwnerScreenState extends State<DashBoardShopOwnerScreen> {  
 @override  
 Widget build(BuildContext context) {  
 return Scaffold(  
 resizeToAvoidBottomInset: false,  
 backgroundColor: backgroundColor,  
 body: Column(  
 children: [  
 Expanded(  
 child: SizedBox(  
 width: 100.h,  
 child: Stack(  
 children: [  
 Container(  
 height: 20.h,  
 width: 100.w,  
 alignment: Alignment.topCenter,  
 padding: EdgeInsets.only(left: 3.h, right: 3.h),  
 decoration: BoxDecoration(  
 color: accentColor,  
 borderRadius: BorderRadius.only(  
 bottomRight: Radius.circular(4.h))),  
 child: Row(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 mainAxisAlignment: MainAxisAlignment.start,  
 children: [  
 GestureDetector(  
 onTap: () {},  
 child: Container(),  
 ),  
 const Spacer(),  
 getCustomTextW6S22(  
 text: 'Dashboard'.tr,  
 color: textWhiteColor,  
 ),  
 const Spacer()  
 ],  
 ),  
 ),  
 Positioned(  
 top: 15.h,  
 right: 0,  
 left: 0,  
 child: Container(  
 width: 100.h,  
 padding:  
 EdgeInsets.symmetric(vertical: 2.h, horizontal: 3.h),  
 alignment: Alignment.topCenter,  
 margin: EdgeInsets.symmetric(horizontal: 3.h),  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 boxShadow: [  
 BoxShadow(  
 color: borderColor,  
 blurRadius: 1,  
 offset: const Offset(  
 0, 2), // changes position of shadow  
 ),  
 ],  
 borderRadius:  
 BorderRadius.all(Radius.circular(10.px))),  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.start,  
 mainAxisAlignment: MainAxisAlignment.spaceBetween,  
 children: [  
 Row(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 mainAxisAlignment: MainAxisAlignment.spaceBetween,  
 children: [  
 Row(  
 crossAxisAlignment: CrossAxisAlignment.end,  
 children: [  
 getSvgImage('app\_logo.svg', height: 4.8.h),  
 getHorSpace(2.h),  
 getCustomTextW6S16(  
 text: 'ALGORTHIMI'.tr, color: accentColor)  
 ],  
 ),  
 getSvgImage('person\_icon.svg', height: 8.3.h),  
 ],  
 ),  
 getCustomTextW6S20(  
 text: 'Muhammad Usman'.tr, color: textBlackColor),  
 Row(  
 children: [  
 getCustomTextW6S16(  
 text: 'Verified Account'.tr,  
 color: greenColor),  
 getHorSpace(1.h),  
 getSvgImage('verify\_account-icon.svg'),  
 ],  
 ),  
 getVerSpace(0.5.h),  
 RichText(  
 text: TextSpan(  
 text: 'Current location: '.tr,  
 style: getCustomTextStyleW4S13(  
 color: subTextColor,  
 ),  
 children: [  
 TextSpan(  
 text: 'Al Barai, Dubai'.tr,  
 style: getCustomTextStyleW10S13(  
 color: subTextColor,  
 ),  
 )  
 ]),  
 ),  
 RichText(  
 text: TextSpan(  
 text: 'Account Type: '.tr,  
 style: getCustomTextStyleW4S13(  
 color: subTextColor,  
 ),  
 children: [  
 TextSpan(  
 text: 'Shop Keeper'.tr,  
 style: getCustomTextStyleW10S13(  
 color: subTextColor,  
 ),  
 )  
 ]),  
 ),  
 RichText(  
 text: TextSpan(  
 text: 'Member Status: '.tr,  
 style: getCustomTextStyleW4S13(  
 color: subTextColor,  
 ),  
 children: [  
 TextSpan(  
 text: 'Bronze'.tr,  
 style: getCustomTextStyleW10S13(  
 color: subTextColor,  
 ),  
 )  
 ]),  
 ),  
 RichText(  
 text: TextSpan(  
 text: 'Questioners: '.tr,  
 style: getCustomTextStyleW4S13(  
 color: subTextColor,  
 ),  
 children: [  
 TextSpan(  
 text: '3/5 Approved'.tr,  
 style: getCustomTextStyleW10S13(  
 color: subTextColor,  
 ),  
 )  
 ]),  
 ),  
 ],  
 ),  
 ),  
 ),  
 ],  
 ),  
 ),  
 ),  
  
  
 Expanded(  
 child: SizedBox(  
 child: Column(children: [  
 GestureDetector(  
 onTap: () {  
 showModalBottomSheet<void>(  
 context: context,  
 elevation: 5,  
 builder: (BuildContext context) {  
 return Container(  
 width: 100.w,  
 height: 37.h,  
 padding: EdgeInsets.symmetric(  
 vertical: 2.h, horizontal: 3.h),  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius: BorderRadius.only(  
 topLeft: Radius.circular(15.px),  
 topRight: Radius.circular(15.px))),  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.start,  
 mainAxisAlignment: MainAxisAlignment.spaceBetween,  
 children: [  
 Column(  
 crossAxisAlignment: CrossAxisAlignment.start,  
 children: [  
 Row(  
 crossAxisAlignment:  
 CrossAxisAlignment.start,  
 mainAxisAlignment:  
 MainAxisAlignment.spaceBetween,  
 children: [  
 getCustomTextW6S17(  
 text: 'Questionnaire'.tr,  
 color: greenColor),  
 InkWell(  
 onTap: () {  
 Get.back();  
 },  
 child: Icon(  
 Icons.close,  
 color: subTextColor,  
 size: 20,  
 ),  
 )  
 ],  
 ),  
 getVerSpace(1.h),  
 getCustomTextW6S12(  
 text:  
 '1) Does the location have a POS system.\n2) # of counters.\n3) Sales of tobacco.\n4) Attach image of tobacco shelf.\n5) Ask shop keep top selling brands.\n6) Sales of energy drinks.'  
 .tr,  
 color: subTextColor,  
 textAlign: TextAlign.start)  
 ],  
 ),  
 Row(  
 children: [  
 Expanded(  
 child: GestureDetector(  
 onTap: () {  
 Get.to(() =>  
 StartQuestionsScreen(  
 userType: 'shopOwner',  
 )  
  
 );  
 },  
 child: fillColorButton(  
 text: 'Accept'.tr,  
 color: pinkAppColor,  
 paddingVertical: 0.7.h),  
 )),  
 getHorSpace(1.h),  
 Expanded(  
 child: GestureDetector(  
 onTap: () {  
 Get.back();  
 },  
 child: outlineButton(  
 borderColor: pinkAppColor,  
 text: 'Cancel'.tr,  
 color: backgroundColor,  
 paddingVertical: 0.7.h),  
 )),  
 ],  
 )  
 ],  
 ),  
 );  
 },  
 );  
 },  
 child: Container(  
 height: 13.h,  
 width: 100.w,  
 margin:  
 EdgeInsets.symmetric(horizontal: 3.h, vertical: 1.h),  
 decoration: BoxDecoration(  
 color: pinkAppColor,  
 borderRadius: BorderRadius.all(Radius.circular(10.px)),  
 ),  
 child: Column(  
 mainAxisAlignment: MainAxisAlignment.center,  
 crossAxisAlignment: CrossAxisAlignment.center,  
 children: [  
 getCustomTextW6S17(  
 text: 'Start New Survey'.tr,  
 color: textWhiteColor,  
 textAlign: TextAlign.center),  
 Padding(  
 padding: EdgeInsets.symmetric(horizontal: 2.h),  
 child: getCustomTextW6S12(  
 text: 'Last Survey Conducted on 26 Nov 2022'.tr,  
 color: textWhiteColor,  
 textAlign: TextAlign.center),  
 )  
 ],  
 ),  
 ),  
 ),  
 GestureDetector(  
 onTap: () {  
 Get.to(() => UploadFileScreen(  
 title: 'Upload Invoices',  
 ));  
 },  
 child: Container(  
 height: 13.h,  
 width: 100.w,  
 margin:  
 EdgeInsets.symmetric(horizontal: 3.h, vertical: 1.h),  
 decoration: BoxDecoration(  
 color: orangeColor,  
 borderRadius: BorderRadius.all(Radius.circular(10.px)),  
 ),  
 child: Column(  
 mainAxisAlignment: MainAxisAlignment.center,  
 crossAxisAlignment: CrossAxisAlignment.center,  
 children: [  
 getCustomTextW6S17(  
 text: 'Upload Invoices'.tr,  
 color: textWhiteColor,  
 textAlign: TextAlign.center),  
 getCustomTextW6S12(  
 text: 'Last Invoice Uploaded on 26 Nov 2022'.tr,  
 color: textWhiteColor,  
 textAlign: TextAlign.center)  
 ],  
 ),  
 ),  
 ),  
 GestureDetector(  
 onTap: () {  
 Get.to(() => UploadFileScreen(  
 title: 'Upload Stock Report',  
 ));  
 },  
 child: Container(  
 height: 13.h,  
 width: 100.w,  
 margin:  
 EdgeInsets.symmetric(horizontal: 3.h, vertical: 1.h),  
 decoration: BoxDecoration(  
 color: lightGreenColor,  
 borderRadius: BorderRadius.all(Radius.circular(10.px)),  
 ),  
 child: Column(  
 mainAxisAlignment: MainAxisAlignment.center,  
 crossAxisAlignment: CrossAxisAlignment.center,  
 children: [  
 getCustomTextW6S17(  
 text: 'Upload Stock Report'.tr,  
 color: textWhiteColor,  
 textAlign: TextAlign.center),  
 getCustomTextW6S12(  
 text: 'Last Report Uploaded on 26 Nov 2022'.tr,  
 color: textWhiteColor,  
 textAlign: TextAlign.center)  
 ],  
 ),  
 ),  
 ),  
 ]),  
 ),  
 ),  
 ],  
 ),  
 );  
 }  
}

# DashboardScreens

## questions\_screen

import 'dart:developer';  
import 'dart:io';  
  
import 'package:algorthimi/utils/data.dart';  
import 'package:algorthimi/view/AuthScreens/welldone\_screen.dart';  
import 'package:algorthimi/view/foot\_step\_counter.dart';  
import 'package:flutter/material.dart';  
import 'package:fluttertoast/fluttertoast.dart';  
import 'package:get/get.dart';  
import 'package:image\_picker/image\_picker.dart';  
import 'package:responsive\_sizer/responsive\_sizer.dart';  
  
import '../../utils/color\_data.dart';  
import '../Widgets/widget\_utils.dart';  
  
//ignore: must\_be\_immutable  
class QuestionsScreen extends StatefulWidget {  
 QuestionsScreen({Key? key, this.userType}) : super(key: key);  
 dynamic userType;  
  
 @override  
 State<QuestionsScreen> createState() => \_QuestionsScreenState();  
}  
  
class \_QuestionsScreenState extends State<QuestionsScreen> {  
 TextEditingController selectBrand1Controller = TextEditingController();  
 TextEditingController selectBrand2Controller = TextEditingController();  
 TextEditingController selectBrand3Controller = TextEditingController();  
 var counter = 0.obs;  
 var yesNoButton = 0.obs;  
 var currentIndex = 0.obs;  
 final ImagePicker imagePicker = ImagePicker();  
 File? takePhoto;  
  
 Future captureImage() async {  
 final pickedFile = await imagePicker.pickImage(source: ImageSource.camera);  
 setState(() {  
 if (pickedFile != null) {  
 takePhoto = File(pickedFile.path);  
 } else {  
 log('No image selected.');  
 }  
 });  
 }  
  
 back() {  
 Get.back();  
 Get.back();  
 Get.back();  
 Get.back();  
 }  
  
 @override  
 Widget build(BuildContext context) {  
 return WillPopScope(  
 onWillPop: () {  
 return back();  
 },  
 child: Scaffold(  
 resizeToAvoidBottomInset: false,  
 backgroundColor: backgroundColor,  
 body: SizedBox(  
 height: 100.h,  
 width: 100.h,  
 child: ListView.builder(  
 shrinkWrap: true,  
 itemCount: DataFile.questionsModel.length,  
 scrollDirection: Axis.horizontal,  
 physics: const NeverScrollableScrollPhysics(),  
 itemBuilder: (context, index) {  
 return Stack(  
 children: [  
 Container(  
 height: 20.h,  
 width: 100.w,  
 alignment: Alignment.topCenter,  
 padding: EdgeInsets.only(top: 6.h, left: 3.h, right: 3.h),  
 decoration: BoxDecoration(  
 color: accentColor,  
 borderRadius: BorderRadius.only(  
 bottomRight: Radius.circular(4.h))),  
 child: Row(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 mainAxisAlignment: MainAxisAlignment.start,  
 children: [  
 GestureDetector(  
 onTap: () {  
 Get.back();  
 Get.back();  
 Get.back();  
 Get.back();  
 },  
 child: Icon(Icons.arrow\_back\_ios\_new\_outlined,  
 color: backgroundColor, size: 2.5.h),  
 ),  
 const Spacer(),  
 Obx(  
 () => getCustomTextW6S22(  
 text:  
 '${'Question'.tr} ${DataFile.questionsModel[currentIndex.value].questionNo}',  
 color: textWhiteColor,  
 ),  
 ),  
 const Spacer()  
 ],  
 ),  
 ),  
 Positioned(  
 top: 15.h,  
 right: 0,  
 left: 0,  
 child: Container(  
 width: 100.h,  
 height: 100.h - 20.h,  
 padding: EdgeInsets.only(top: 3.h),  
 alignment: Alignment.topCenter,  
 margin: EdgeInsets.symmetric(horizontal: 3.h),  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius:  
 BorderRadius.all(Radius.circular(10.px))),  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 mainAxisAlignment: MainAxisAlignment.spaceBetween,  
 children: [  
 Padding(  
 padding: EdgeInsets.symmetric(horizontal: 3.h),  
 child: Column(  
 children: [  
 Obx(  
 () => getCustomTextW6S15(  
 text: DataFile  
 .questionsModel[currentIndex.value]  
 .questionTitle  
 .toString()  
 .tr,  
 color: textBlackColor,  
 textAlign: TextAlign.center),  
 ),  
 ],  
 ),  
 ),  
 takePhoto == null ||  
 DataFile.questionsModel[currentIndex.value]  
 .questionType !=  
 '3'  
 ? Column(  
 children: [  
 getVerSpace(10.h),  
 Stack(  
 alignment: Alignment.center,  
 children: [  
 getSvgImage('back\_logo.svg',  
 height: 25.h),  
 Obx(  
 () => Padding(  
 padding: EdgeInsets.symmetric(  
 horizontal: 9.h),  
 child: DataFile  
 .questionsModel[  
 currentIndex.value]  
 .questionType ==  
 '3'  
 ? GestureDetector(  
 onTap: () {  
 captureImage();  
 },  
 child: outlineButton(  
 text: 'Take Picture'.tr,  
 borderColor:  
 subTextColor,  
 color: backgroundColor),  
 )  
 : DataFile  
 .questionsModel[  
 currentIndex  
 .value]  
 .questionType ==  
 '2'  
 ? Column(  
 children: [  
 Obx(  
 () => outlineButton(  
 text: counter  
 .value  
 .toString(),  
 color:  
 backgroundColor,  
 borderColor:  
 pinkAppColor),  
 ),  
 getVerSpace(2.h),  
 Row(  
 children: [  
 Expanded(  
 child:  
 GestureDetector(  
 onTap: () {  
 if (counter  
 .value >  
 0) {  
 counter  
 .value--;  
 }  
 },  
 child: outlineButton(  
 text: '-',  
 color:  
 backgroundColor,  
 borderColor:  
 subTextColor),  
 )),  
 getHorSpace(2.h),  
 Expanded(  
 child:  
 GestureDetector(  
 onTap: () {  
 counter  
 .value++;  
 },  
 child: outlineButton(  
 text: '+',  
 color:  
 backgroundColor,  
 borderColor:  
 subTextColor),  
 )),  
 ],  
 ),  
 ],  
 )  
 : DataFile  
 .questionsModel[  
 currentIndex  
 .value]  
 .questionType ==  
 '1'  
 ? Obx(  
 () => Column(  
 children: [  
 GestureDetector(  
 onTap: () {  
 yesNoButton  
 .value = 1;  
 },  
 child: outlineButton(  
 text: 'Yes'  
 .tr,  
 color: yesNoButton.value ==  
 1  
 ? pinkAppColor  
 : backgroundColor,  
 borderColor:  
 pinkAppColor,  
 textColor: yesNoButton.value ==  
 1  
 ? textWhiteColor  
 : pinkAppColor),  
 ),  
 getVerSpace(  
 2.h),  
 GestureDetector(  
 onTap: () {  
 yesNoButton  
 .value = 2;  
 },  
 child: outlineButton(  
 text: 'No'  
 .tr,  
 color: yesNoButton.value ==  
 2  
 ? accentColor  
 : backgroundColor,  
 borderColor:  
 accentColor,  
 textColor: yesNoButton.value ==  
 2  
 ? textWhiteColor  
 : accentColor),  
 ),  
 ],  
 ),  
 )  
 : Column(  
 children: [  
 Container(  
 decoration: BoxDecoration(  
 color:  
 backgroundColor,  
 borderRadius:  
 BorderRadius.all(  
 Radius.circular(15.px)),  
 boxShadow: [  
 BoxShadow(  
 color:  
 borderColor,  
 blurRadius:  
 10,  
 offset: const Offset(  
 0,  
 0), // changes position of shadow  
 ),  
 ]),  
 child:  
 getCustomTextFormField(  
 keyboardType:  
 TextInputType  
 .text,  
 obscureText:  
 false,  
 readOnly:  
 true,  
 hintText:  
 'Select Brand'  
 .tr,  
 controller:  
 selectBrand1Controller,  
 suffixIcon: Icon(  
 Icons  
 .arrow\_drop\_down\_rounded,  
 color:  
 subTextColor),  
 onTap:  
 () {  
 showModalBottomSheet<  
 void>(  
 context:  
 context,  
 elevation:  
 5,  
 builder:  
 (BuildContext context) {  
 return Container(  
 width: 100.w,  
 height: 25.h,  
 padding: EdgeInsets.symmetric(vertical: 2.h),  
 decoration: BoxDecoration(color: backgroundColor, borderRadius: BorderRadius.only(topLeft: Radius.circular(15.px), topRight: Radius.circular(15.px))),  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.start,  
 mainAxisAlignment: MainAxisAlignment.spaceEvenly,  
 children: [  
 GestureDetector(  
 onTap: () {  
 selectBrand1Controller.text = 'Nike'.toString().tr;  
 Get.back();  
 },  
 child: Container(  
 width: 100.w,  
 alignment: Alignment.center,  
 margin: EdgeInsets.symmetric(horizontal: 3.h),  
 decoration: BoxDecoration(  
 border: Border.all(color: textBlackColor),  
 borderRadius: BorderRadius.all(  
 Radius.circular(10.px),  
 )),  
 padding: EdgeInsets.symmetric(vertical: 1.h),  
 child: getCustomTextW6S15(  
 text: 'Nike'.tr,  
 color: textBlackColor,  
 ),  
 ),  
 ),  
 GestureDetector(  
 onTap: () {  
 selectBrand1Controller.text = 'Hermes'.toString().tr;  
 Get.back();  
 },  
 child: Container(  
 width: 100.w,  
 alignment: Alignment.center,  
 margin: EdgeInsets.symmetric(horizontal: 3.h),  
 decoration: BoxDecoration(  
 border: Border.all(color: textBlackColor),  
 borderRadius: BorderRadius.all(  
 Radius.circular(10.px),  
 )),  
 padding: EdgeInsets.symmetric(vertical: 1.h),  
 child: getCustomTextW6S15(  
 text: 'Hermes'.tr,  
 color: textBlackColor,  
 ),  
 ),  
 ),  
 GestureDetector(  
 onTap: () {  
 selectBrand1Controller.text = 'Louis Vuitton'.toString().tr;  
 Get.back();  
 },  
 child: Container(  
 width: 100.w,  
 alignment: Alignment.center,  
 margin: EdgeInsets.symmetric(horizontal: 3.h),  
 decoration: BoxDecoration(  
 border: Border.all(color: textBlackColor),  
 borderRadius: BorderRadius.all(  
 Radius.circular(10.px),  
 )),  
 padding: EdgeInsets.symmetric(vertical: 1.h),  
 child: getCustomTextW6S15(  
 text: 'Louis Vuitton'.tr,  
 color: textBlackColor,  
 ),  
 ),  
 ),  
 ],  
 ),  
 );  
 },  
 );  
 }),  
 ),  
 getVerSpace(  
 1.5.h),  
 Container(  
 decoration: BoxDecoration(  
 color:  
 backgroundColor,  
 borderRadius:  
 BorderRadius.all(  
 Radius.circular(15.px)),  
 boxShadow: [  
 BoxShadow(  
 color:  
 borderColor,  
 blurRadius:  
 10,  
 offset: const Offset(  
 0,  
 0), // changes position of shadow  
 ),  
 ]),  
 child:  
 getCustomTextFormField(  
 keyboardType:  
 TextInputType  
 .text,  
 obscureText:  
 false,  
 readOnly:  
 true,  
 hintText:  
 'Select Brand'  
 .tr,  
 controller:  
 selectBrand2Controller,  
 suffixIcon: Icon(  
 Icons  
 .arrow\_drop\_down\_rounded,  
 color:  
 subTextColor),  
 onTap:  
 () {  
 showModalBottomSheet<  
 void>(  
 context:  
 context,  
 elevation:  
 5,  
 builder:  
 (BuildContext context) {  
 return Container(  
 width: 100.w,  
 height: 25.h,  
 padding: EdgeInsets.symmetric(vertical: 2.h),  
 decoration: BoxDecoration(color: backgroundColor, borderRadius: BorderRadius.only(topLeft: Radius.circular(15.px), topRight: Radius.circular(15.px))),  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.start,  
 mainAxisAlignment: MainAxisAlignment.spaceEvenly,  
 children: [  
 GestureDetector(  
 onTap: () {  
 selectBrand2Controller.text = 'Nike'.toString().tr;  
 Get.back();  
 },  
 child: Container(  
 width: 100.w,  
 alignment: Alignment.center,  
 margin: EdgeInsets.symmetric(horizontal: 3.h),  
 decoration: BoxDecoration(  
 border: Border.all(color: textBlackColor),  
 borderRadius: BorderRadius.all(  
 Radius.circular(10.px),  
 )),  
 padding: EdgeInsets.symmetric(vertical: 1.h),  
 child: getCustomTextW6S15(  
 text: 'Nike'.tr,  
 color: textBlackColor,  
 ),  
 ),  
 ),  
 GestureDetector(  
 onTap: () {  
 selectBrand2Controller.text = 'Hermes'.toString().tr;  
 Get.back();  
 },  
 child: Container(  
 width: 100.w,  
 alignment: Alignment.center,  
 margin: EdgeInsets.symmetric(horizontal: 3.h),  
 decoration: BoxDecoration(  
 border: Border.all(color: textBlackColor),  
 borderRadius: BorderRadius.all(  
 Radius.circular(10.px),  
 )),  
 padding: EdgeInsets.symmetric(vertical: 1.h),  
 child: getCustomTextW6S15(  
 text: 'Hermes'.tr,  
 color: textBlackColor,  
 ),  
 ),  
 ),  
 GestureDetector(  
 onTap: () {  
 selectBrand2Controller.text = 'Louis Vuitton'.toString().tr;  
 Get.back();  
 },  
 child: Container(  
 width: 100.w,  
 alignment: Alignment.center,  
 margin: EdgeInsets.symmetric(horizontal: 3.h),  
 decoration: BoxDecoration(  
 border: Border.all(color: textBlackColor),  
 borderRadius: BorderRadius.all(  
 Radius.circular(10.px),  
 )),  
 padding: EdgeInsets.symmetric(vertical: 1.h),  
 child: getCustomTextW6S15(  
 text: 'Louis Vuitton'.tr,  
 color: textBlackColor,  
 ),  
 ),  
 ),  
 ],  
 ),  
 );  
 },  
 );  
 }),  
 ),  
 getVerSpace(  
 1.5.h),  
 Container(  
 decoration: BoxDecoration(  
 color:  
 backgroundColor,  
 borderRadius:  
 BorderRadius.all(  
 Radius.circular(15.px)),  
 boxShadow: [  
 BoxShadow(  
 color:  
 borderColor,  
 blurRadius:  
 10,  
 offset: const Offset(  
 0,  
 0), // changes position of shadow  
 ),  
 ]),  
 child:  
 getCustomTextFormField(  
 keyboardType:  
 TextInputType  
 .text,  
 obscureText:  
 false,  
 readOnly:  
 true,  
 hintText:  
 'Select Brand'  
 .tr,  
 controller:  
 selectBrand3Controller,  
 suffixIcon: Icon(  
 Icons  
 .arrow\_drop\_down\_rounded,  
 color:  
 subTextColor),  
 onTap:  
 () {  
 showModalBottomSheet<  
 void>(  
 context:  
 context,  
 elevation:  
 5,  
 builder:  
 (BuildContext context) {  
 return Container(  
 width: 100.w,  
 height: 25.h,  
 padding: EdgeInsets.symmetric(vertical: 2.h),  
 decoration: BoxDecoration(color: backgroundColor, borderRadius: BorderRadius.only(topLeft: Radius.circular(15.px), topRight: Radius.circular(15.px))),  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.start,  
 mainAxisAlignment: MainAxisAlignment.spaceEvenly,  
 children: [  
 GestureDetector(  
 onTap: () {  
 selectBrand3Controller.text = 'Nike'.toString().tr;  
 Get.back();  
 },  
 child: Container(  
 width: 100.w,  
 alignment: Alignment.center,  
 margin: EdgeInsets.symmetric(horizontal: 3.h),  
 decoration: BoxDecoration(  
 border: Border.all(color: textBlackColor),  
 borderRadius: BorderRadius.all(  
 Radius.circular(10.px),  
 )),  
 padding: EdgeInsets.symmetric(vertical: 1.h),  
 child: getCustomTextW6S15(  
 text: 'Nike'.tr,  
 color: textBlackColor,  
 ),  
 ),  
 ),  
 GestureDetector(  
 onTap: () {  
 selectBrand3Controller.text = 'Hermes'.toString().tr;  
 Get.back();  
 },  
 child: Container(  
 width: 100.w,  
 alignment: Alignment.center,  
 margin: EdgeInsets.symmetric(horizontal: 3.h),  
 decoration: BoxDecoration(  
 border: Border.all(color: textBlackColor),  
 borderRadius: BorderRadius.all(  
 Radius.circular(10.px),  
 )),  
 padding: EdgeInsets.symmetric(vertical: 1.h),  
 child: getCustomTextW6S15(  
 text: 'Hermes'.tr,  
 color: textBlackColor,  
 ),  
 ),  
 ),  
 GestureDetector(  
 onTap: () {  
 selectBrand3Controller.text = 'Louis Vuitton'.toString().tr;  
 Get.back();  
 },  
 child: Container(  
 width: 100.w,  
 alignment: Alignment.center,  
 margin: EdgeInsets.symmetric(horizontal: 3.h),  
 decoration: BoxDecoration(  
 border: Border.all(color: textBlackColor),  
 borderRadius: BorderRadius.all(  
 Radius.circular(10.px),  
 )),  
 padding: EdgeInsets.symmetric(vertical: 1.h),  
 child: getCustomTextW6S15(  
 text: 'Louis Vuitton'.tr,  
 color: textBlackColor,  
 ),  
 ),  
 ),  
 ],  
 ),  
 );  
 },  
 );  
 }),  
 ),  
 ],  
 ),  
 ),  
 ),  
 ],  
 ),  
 getVerSpace(20.h),  
 ],  
 )  
 : Expanded(  
 child: Column(  
 children: [  
 Expanded(  
 child: Container(  
 height: 50.h,  
 margin: EdgeInsets.symmetric(  
 vertical: 3.h),  
 decoration: BoxDecoration(  
 color: borderColor,  
 borderRadius: BorderRadius.all(  
 Radius.circular(10.px)),  
 image: DecorationImage(  
 image: FileImage(takePhoto!),  
 fit: BoxFit.cover)),  
 ),  
 ),  
 GestureDetector(  
 onTap: () {  
 captureImage();  
 },  
 child: outlineButton(  
 text: 'Take Picture'.tr,  
 borderColor: subTextColor,  
 color: backgroundColor),  
 ),  
 getVerSpace(1.h),  
 ],  
 ),  
 ),  
 GestureDetector(  
 onTap: () {  
 setState(() {});  
 if (DataFile.questionsModel[currentIndex.value]  
 .questionType ==  
 '1') {  
 if (yesNoButton.value != 0) {  
 yesNoButton.value = 0;  
 if (currentIndex.value <  
 DataFile.questionsModel.length - 1) {  
 currentIndex.value++;  
 } else {  
 Get.to(() => const FootStepCounter());  
 // Get.to(() => WellDoneScreen(  
 // screenTitle: 'question',  
 // userType: widget.userType,  
 // )  
 // );  
 }  
 } else {  
 Fluttertoast.showToast(  
 msg: 'Select Option'.tr,  
 toastLength: Toast.LENGTH\_SHORT,  
 gravity: ToastGravity.TOP,  
 timeInSecForIosWeb: 1,  
 backgroundColor: pinkAppColor,  
 textColor: textWhiteColor,  
 fontSize: 14.0);  
 }  
 } else if (DataFile  
 .questionsModel[currentIndex.value]  
 .questionType ==  
 '2') {  
 if (counter.value != 0) {  
 if (currentIndex.value <  
 DataFile.questionsModel.length - 1) {  
 currentIndex.value++;  
 } else {  
 Get.to(() => const FootStepCounter());  
 // Get.to(() => WellDoneScreen(  
 // screenTitle: 'question',  
 // userType: widget.userType,  
 // )  
 // );  
 }  
 } else {  
 Fluttertoast.showToast(  
 msg: 'Select Amount'.tr,  
 toastLength: Toast.LENGTH\_SHORT,  
 gravity: ToastGravity.TOP,  
 timeInSecForIosWeb: 1,  
 backgroundColor: pinkAppColor,  
 textColor: textWhiteColor,  
 fontSize: 14.0);  
 }  
 } else if (DataFile  
 .questionsModel[currentIndex.value]  
 .questionType ==  
 '3') {  
 if (takePhoto != null) {  
 if (currentIndex.value <  
 DataFile.questionsModel.length - 1) {  
 currentIndex.value++;  
 } else {  
 Get.to(() => const FootStepCounter());  
 // Get.to(() => WellDoneScreen(  
 // screenTitle: 'question',  
 // userType: widget.userType,  
 // ));  
 }  
 } else {  
 Fluttertoast.showToast(  
 msg: 'Upload Image'.tr,  
 toastLength: Toast.LENGTH\_SHORT,  
 gravity: ToastGravity.TOP,  
 timeInSecForIosWeb: 1,  
 backgroundColor: pinkAppColor,  
 textColor: textWhiteColor,  
 fontSize: 14.0);  
 }  
 } else if (DataFile  
 .questionsModel[currentIndex.value]  
 .questionType ==  
 '4') {  
 if (selectBrand1Controller.text.isNotEmpty &&  
 selectBrand2Controller.text.isNotEmpty &&  
 selectBrand3Controller.text.isNotEmpty) {  
 if (currentIndex.value <  
 DataFile.questionsModel.length - 1) {  
 currentIndex.value++;  
 } else {  
 Get.to(() => const FootStepCounter());  
 // Get.to(() => WellDoneScreen(  
 // screenTitle: 'question',  
 // userType: widget.userType,  
 // ));  
 }  
 } else {  
 Fluttertoast.showToast(  
 msg: 'Select Brand'.tr,  
 toastLength: Toast.LENGTH\_SHORT,  
 gravity: ToastGravity.TOP,  
 timeInSecForIosWeb: 1,  
 backgroundColor: pinkAppColor,  
 textColor: textWhiteColor,  
 fontSize: 14.0);  
 }  
 }  
 },  
 child: fillColorButton(  
 color: pinkAppColor, text: 'Next'.tr),  
 )  
 ],  
 ),  
 ),  
 ),  
 ],  
 );  
 },  
 ),  
 ),  
 ),  
 );  
 }  
}

# DashboardScreens

## start\_questions\_screen

import 'dart:async';  
import 'dart:developer';  
import 'package:algorthimi/view/DashboardScreens/questions\_screen.dart';  
import 'package:flutter/material.dart';  
import 'package:get/get.dart';  
import 'package:pedometer/pedometer.dart';  
import 'package:responsive\_sizer/responsive\_sizer.dart';  
  
import '../../utils/color\_data.dart';  
import '../AuthScreens/take\_picture\_screen.dart';  
import '../Widgets/widget\_utils.dart';  
  
//ignore: must\_be\_immutable  
class StartQuestionsScreen extends StatefulWidget {  
 StartQuestionsScreen({Key? key, this.userType}) : super(key: key);  
 dynamic userType;  
  
 @override  
 State<StartQuestionsScreen> createState() => \_StartQuestionsScreenState();  
}  
  
class \_StartQuestionsScreenState extends State<StartQuestionsScreen> {  
 final interval = const Duration(seconds: 1);  
 final int timerMaxSeconds = 60 \* 60;  
 var currentSeconds = 0.obs;  
 Timer? surveyTimer;  
  
 String get timerText =>  
 '${((timerMaxSeconds - currentSeconds.value) ~/ 60).toString().padLeft(2, '0')} : ${((timerMaxSeconds - currentSeconds.value) % 60).toString().padLeft(2, '0')}';  
  
 startTimeout() {  
 var duration = interval;  
 surveyTimer = Timer.periodic(duration, (timer) {  
 log(timer.tick.toString());  
 currentSeconds.value = timer.tick;  
 if (timer.tick >= timerMaxSeconds) timer.cancel();  
 });  
 }  
  
  
 //////////////////////////////////  
  
 late Stream<StepCount> \_stepCountStream;  
 int \_totalSteps = 0;  
 bool \_isCountingSteps = false;  
 final Stopwatch \_stopwatch = Stopwatch();  
 late Timer \_timer;  
  
 // Conversion factor: 1 meter^2 = 10.7639 square feet  
 final double squareFeetPerSquareMeter = 10.7639;  
 // Conversion factor: 1 square meter = 0.005 marlas  
 final double marlasPerSquareMeter = 0.005;  
  
 // Assuming an average stride length of 0.75 meters  
 final double averageStrideLengthMeters = 0.75;  
  
 @override  
 void initState() {  
 startTimeout();  
 super.initState();  
 \_initPedometer();  
 }  
  
 void \_initPedometer() {  
 \_stepCountStream = Pedometer.stepCountStream;  
 \_stepCountStream.listen((StepCount event) {  
 if (\_isCountingSteps) {  
 setState(() {  
 \_totalSteps = event.steps;  
 });  
 }  
 });  
 }  
  
 void \_startCountingSteps() {  
 setState(() {  
 \_isCountingSteps = true;  
 \_stopwatch.start();  
 \_startTimer();  
 });  
 }  
  
 void \_stopCountingSteps() {  
 setState(() {  
 \_isCountingSteps = false;  
 \_stopwatch.stop();  
 \_stopTimer();  
 });  
 }  
  
 void \_resetTime() {  
 setState(() {  
 \_stopwatch.reset();  
 \_totalSteps = 0;  
 });  
 }  
  
 void \_startTimer() {  
 \_timer = Timer.periodic(Duration(seconds: 1), (timer) {  
 setState(() {});  
 });  
 }  
  
 void \_stopTimer() {  
 \_timer.cancel();  
 }  
  
 String \_formatElapsedTime(Duration duration) {  
 final minutes = duration.inMinutes;  
 final seconds = duration.inSeconds % 60;  
 return '$minutes:${seconds.toString().padLeft(2, '0')}';  
 }  
  
 double \_calculateDistanceInSquareMeters(int steps) {  
 return steps \* averageStrideLengthMeters;  
 }  
  
 double \_convertSquareMetersToSquareFeet(double squareMeters) {  
 return squareMeters \* squareFeetPerSquareMeter;  
 }  
  
 double \_convertSquareMetersToMarlas(double squareMeters) {  
 return squareMeters \* marlasPerSquareMeter;  
 }  
  
  
 @override  
 void dispose() {  
 surveyTimer!.cancel();  
 super.dispose();  
 }  
  
 back() {  
 Get.back();  
 Get.back();  
 }  
  
 @override  
 Widget build(BuildContext context) {  
  
 double distanceInSquareMeters = \_calculateDistanceInSquareMeters(\_totalSteps);  
 double distanceInSquareFeet = \_convertSquareMetersToSquareFeet(distanceInSquareMeters);  
 double distanceInMarlas = \_convertSquareMetersToMarlas(distanceInSquareMeters);  
  
  
  
 return Obx(  
 () => Scaffold(  
 resizeToAvoidBottomInset: false,  
 backgroundColor: backgroundColor,  
 body: WillPopScope(  
 onWillPop: () {  
 return back();  
 },  
 child: SingleChildScrollView(  
 child: Column(  
 mainAxisAlignment: MainAxisAlignment.spaceBetween,  
 children: [  
 Container(  
 width: 100.h,  
 height: 85.h,  
 padding: EdgeInsets.only(  
 top: 20.h, right: 3.h, left: 3.h, bottom: 2.h),  
 decoration: BoxDecoration(  
 color: accentColor,  
 borderRadius:  
 BorderRadius.only(bottomRight: Radius.circular(20.h))),  
 child: Padding(  
 padding: EdgeInsets.symmetric(vertical: 2.h),  
 child: SingleChildScrollView(  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 mainAxisAlignment: MainAxisAlignment.spaceBetween,  
 children: [  
 getCustomTextW6S20(  
 text: 'Let’s Start...'.tr, color: textWhiteColor),  
 SizedBox(height: 20,),  
 Column(  
 children: [  
 getCustomTextW6S17(  
 text: 'Quick Supermarket, Majan'.tr,  
 color: textWhiteColor),  
 getVerSpace(3.h),  
 getCustomTextW6S17(  
 text: 'Your slot has been blocked for\n60 min'.tr,  
 color: textWhiteColor,  
 textAlign: TextAlign.center),  
 ],  
 ),  
 SizedBox(height: 60,),  
 timerButton(  
 text: timerText,  
 ),  
  
 ],  
 ),  
 ),  
 ),  
 ),  
 SizedBox(height: 40,),  
 Padding(  
 padding: EdgeInsets.only(bottom: 4.h, left: 3.h, right: 3.h),  
 child: GestureDetector(  
 onTap: () {  
 widget.userType.toString() == 'freelancer'  
 ? Get.to(() => TakePictureScreen(  
 isSelectSignUp: 2,  
 userType: widget.userType,  
 ))  
 : Get.to(() => TakePictureScreen(  
 isSelectSignUp: 3,  
 userType: widget.userType,  
 ));  
 },  
 child: fillColorButton(  
 text: 'Start Survey Now!'.tr, color: pinkAppColor),  
 ),  
 )  
 ],  
 ),  
 ),  
 ),  
 ),  
 );  
 }  
}

# DashboardScreens

## upload\_file\_screen

import 'package:file\_picker/file\_picker.dart';  
import 'package:flutter/material.dart';  
import 'package:fluttertoast/fluttertoast.dart';  
import 'package:get/get.dart';  
import 'package:responsive\_sizer/responsive\_sizer.dart';  
  
import '../../utils/color\_data.dart';  
import '../AuthScreens/welldone\_screen.dart';  
import '../Widgets/widget\_utils.dart';  
  
//ignore: must\_be\_immutable  
class UploadFileScreen extends StatefulWidget {  
 UploadFileScreen({Key? key, this.title}) : super(key: key);  
 dynamic title;  
  
 @override  
 State<UploadFileScreen> createState() => \_UploadFileScreenState();  
}  
  
class \_UploadFileScreenState extends State<UploadFileScreen> {  
 FilePickerResult? selectedFile;  
  
 void selectFile() async {  
 selectedFile = await FilePicker.platform.pickFiles(  
 type: FileType.custom,  
 allowedExtensions: ['xlsx'],  
 );  
 setState(() {});  
 }  
  
 @override  
 Widget build(BuildContext context) {  
 return Scaffold(  
 resizeToAvoidBottomInset: false,  
 backgroundColor: backgroundColor,  
 body: SizedBox(  
 height: 100.h,  
 width: 100.h,  
 child: Stack(  
 children: [  
 Container(  
 height: 20.h,  
 width: 100.w,  
 alignment: Alignment.topCenter,  
 padding: EdgeInsets.only(top: 6.h, left: 3.h, right: 3.h),  
 decoration: BoxDecoration(  
 color: accentColor,  
 borderRadius:  
 BorderRadius.only(bottomRight: Radius.circular(4.h))),  
 child: Row(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 mainAxisAlignment: MainAxisAlignment.start,  
 children: [  
 GestureDetector(  
 onTap: () {  
 Get.back();  
 },  
 child: Icon(Icons.arrow\_back\_ios\_new\_outlined,  
 color: backgroundColor, size: 2.5.h),  
 ),  
  
 const Spacer(),  
   
 getCustomTextW6S22(  
 text: widget.title.toString().tr,  
 color: textWhiteColor,  
 ),  
 const Spacer()  
 ],  
 ),  
 ),  
 Positioned(  
 top: 15.h,  
 right: 0,  
 left: 0,  
 child: Container(  
 width: 100.h,  
 height: 100.h - 20.h,  
 padding: EdgeInsets.only(top: 2.h),  
 alignment: Alignment.topCenter,  
 margin: EdgeInsets.symmetric(horizontal: 3.h),  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius: BorderRadius.all(Radius.circular(10.px))),  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 mainAxisAlignment: MainAxisAlignment.spaceBetween,  
 children: [  
 getCustomTextW6S15(  
 text: '',  
 color: textBlackColor,  
 textAlign: TextAlign.center),  
 Stack(  
 alignment: Alignment.center,  
 children: [  
 getSvgImage('back\_logo.svg', height: 29.h),  
 selectedFile != null  
 ? SizedBox(  
 height: 30.h,  
 child: Column(  
 mainAxisAlignment:  
 MainAxisAlignment.spaceBetween,  
 crossAxisAlignment: CrossAxisAlignment.center,  
 children: [  
 Column(  
 children: [  
 getSvgImage('excel\_icon.svg',  
 height: 15.h),  
 getVerSpace(1.h),  
 getCustomTextW6S15(  
 text: selectedFile!.files.first.name  
 .toString(),  
 color: textBlackColor,  
 textAlign: TextAlign.center)  
 ],  
 ),  
 GestureDetector(  
 onTap: () {  
 selectFile();  
 },  
 child: Padding(  
 padding: EdgeInsets.symmetric(  
 horizontal: 9.h),  
 child: outlineButton(  
 text: 'Upload File'.tr,  
 borderColor: subTextColor,  
 color: backgroundColor),  
 ),  
 ),  
 ],  
 ),  
 )  
 : GestureDetector(  
 onTap: () {  
 selectFile();  
 },  
 child: Padding(  
 padding:  
 EdgeInsets.symmetric(horizontal: 9.h),  
 child: outlineButton(  
 text: 'Upload File'.tr,  
 borderColor: subTextColor,  
 color: backgroundColor),  
 ),  
 ),  
 ],  
 ),  
 GestureDetector(  
 onTap: () {  
 if (selectedFile != null) {  
 Get.to(() => WellDoneScreen(  
 screenTitle: 'uploadFile',  
 userType: 'shopOwner',  
 ));  
 } else {  
 Fluttertoast.showToast(  
 msg: 'Select File'.tr,  
 toastLength: Toast.LENGTH\_SHORT,  
 gravity: ToastGravity.TOP,  
 timeInSecForIosWeb: 1,  
 backgroundColor: pinkAppColor,  
 textColor: textWhiteColor,  
 fontSize: 14.0);  
 }  
 },  
 child: fillColorButton(color: pinkAppColor, text: 'Next'.tr),  
 ),  
 ],  
 ),  
 ),  
 ),  
 ],  
 ),  
 ),  
 );  
 }  
}

# SettingScreens

## accounts\_screen

import 'package:flutter/material.dart';  
import 'package:get/get.dart';  
import 'package:responsive\_sizer/responsive\_sizer.dart';  
  
import '../../utils/color\_data.dart';  
import '../WalletScreens/add\_card.dart';  
import '../WalletScreens/select\_bank.dart';  
import '../WalletScreens/wallet\_screen.dart';  
import '../Widgets/widget\_utils.dart';  
  
class AccountsScreen extends StatefulWidget {  
 const AccountsScreen({Key? key}) : super(key: key);  
  
 @override  
 State<AccountsScreen> createState() => \_AccountsScreenState();  
}  
  
class \_AccountsScreenState extends State<AccountsScreen> {  
 var selectedPaymentMethod = 0.obs;  
  
 @override  
 Widget build(BuildContext context) {  
 return Scaffold(  
 resizeToAvoidBottomInset: false,  
 backgroundColor: backgroundColor,  
 body: SizedBox(  
 width: 100.w,  
 child: Column(  
 children: [  
 SizedBox(  
 width: 100.h,  
 child: Stack(  
 children: [  
 Container(  
 height: 20.h,  
 width: 100.w,  
 alignment: Alignment.center,  
 padding: EdgeInsets.only(left: 3.h, right: 3.h),  
 decoration: BoxDecoration(  
 color: accentColor,  
 borderRadius: BorderRadius.only(  
 bottomRight: Radius.circular(4.h))),  
 child: Row(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 mainAxisAlignment: MainAxisAlignment.start,  
 children: [  
 GestureDetector(  
 onTap: () {  
 Get.back();  
 },  
 child: Icon(Icons.arrow\_back\_ios\_new\_outlined,  
 color: backgroundColor, size: 2.5.h),  
 ),  
 const Spacer(),  
 getCustomTextW6S22(  
 text: 'Accounts'.tr,  
 color: textWhiteColor,  
 ),  
 const Spacer()  
 ],  
 ),  
 ),  
 Positioned(  
 top: 15.h,  
 right: 0,  
 left: 0,  
 child: Container(  
 width: 100.h,  
 height: 5.h,  
 padding:  
 EdgeInsets.symmetric(vertical: 2.h, horizontal: 3.h),  
 alignment: Alignment.topCenter,  
 margin: EdgeInsets.symmetric(horizontal: 3.h),  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius: BorderRadius.only(  
 topRight: Radius.circular(10.px),  
 topLeft: Radius.circular(10.px)),  
 ),  
 ),  
 ),  
 ],  
 ),  
 ),  
 getVerSpace(1.h),  
 Expanded(  
 child: SizedBox(  
 width: 100.w,  
 child: Padding(  
 padding: EdgeInsets.symmetric(horizontal: 4.0.h),  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.start,  
 children: [  
 getCustomTextW6S16(  
 text: 'Saved Payment Methods'.tr, color: accentColor),  
 getVerSpace(2.h),  
 Expanded(  
 child: SizedBox(  
 child: ListView.builder(  
 padding: EdgeInsets.zero,  
 itemCount: withDrawTransferTitle.length,  
 shrinkWrap: true,  
 physics: const BouncingScrollPhysics(),  
 itemBuilder: (context, index) {  
 return Container(  
 height: 7.h,  
 width: 100.w,  
 padding: EdgeInsets.symmetric(horizontal: 3.h),  
 margin: EdgeInsets.symmetric(  
 vertical: 0.5.h, horizontal: 1.h),  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius:  
 BorderRadius.all(Radius.circular(10.px)),  
 boxShadow: [  
 BoxShadow(  
 color: borderColor,  
 blurRadius: 2,  
 offset: const Offset(  
 0, 0), // changes position of shadow  
 ),  
 ],  
 ),  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.start,  
 mainAxisAlignment: MainAxisAlignment.center,  
 children: [  
 getCustomTextW6S15(  
 text: withDrawTransferTitle[index].toString().tr,maxLines: 1,  
 color: textBlackColor,),  
 getCustomTextW6S12(  
 text: withDrawTransferSubTitle[index].toString().tr,  
 color: subTextColor),  
 ],  
 ),  
 );  
 },  
 ),  
 ),  
 ),  
 GestureDetector(  
 onTap: () {  
 showModalBottomSheet<void>(  
 context: context,  
 elevation: 5,  
 builder: (BuildContext context) {  
 return Container(  
 height: 38.h,  
 width: 100.w,  
 padding: EdgeInsets.only(  
 right: 3.h,  
 left: 3.h,  
 bottom: 4.h,  
 top: 2.h),  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius: BorderRadius.only(  
 topLeft: Radius.circular(15.px),  
 topRight: Radius.circular(15.px))),  
 child: Column(  
 crossAxisAlignment:  
 CrossAxisAlignment.start,  
 mainAxisAlignment:  
 MainAxisAlignment.spaceBetween,  
 children: [  
 Column(  
 children: [  
 Row(  
 crossAxisAlignment:  
 CrossAxisAlignment.center,  
 mainAxisAlignment:  
 MainAxisAlignment.spaceBetween,  
 children: [  
 getCustomTextW6S20(  
 text: 'Add Payment Method'.tr,  
 color: accentColor),  
 InkWell(  
 onTap: () {  
 Get.back();  
 },  
 child: Icon(  
 Icons.close,  
 color: subTextColor,  
 size: 20,  
 ),  
 )  
 ],  
 ),  
 ListView.builder(  
 shrinkWrap: true,  
 itemCount: paymentMethodList.length,  
 padding: EdgeInsets.symmetric(  
 vertical: 2.h),  
 physics:  
 const NeverScrollableScrollPhysics(),  
 itemBuilder: (context, index) {  
 return GestureDetector(  
 onTap: () {  
 selectedPaymentMethod.value =  
 index;  
 },  
 child: Obx(  
 () => Container(  
 color: backgroundColor,  
 margin: EdgeInsets.only(  
 bottom: 2.h),  
 child: Column(  
 crossAxisAlignment:  
 CrossAxisAlignment  
 .start,  
 children: [  
 Row(  
 crossAxisAlignment:  
 CrossAxisAlignment  
 .center,  
 mainAxisAlignment:  
 MainAxisAlignment  
 .start,  
 children: [  
 Container(  
 width: 1.8.h,  
 height: 1.8.h,  
 decoration: BoxDecoration(  
 shape: BoxShape  
 .circle,  
 border: Border.all(  
 color: selectedPaymentMethod.value ==  
 index  
 ? accentColor  
 : borderColor,  
 width: selectedPaymentMethod.value ==  
 index  
 ? 4.px  
 : 1.5  
 .px)),  
 ),  
 getHorSpace(2.h),  
 getCustomTextW6S16(  
 text:  
 paymentMethodList[  
 index]  
 .toString()  
 .tr,  
 color:  
 subTextColor,  
 ),  
 ],  
 ),  
 ],  
 ),  
 ),  
 ),  
 );  
 },  
 ),  
 ],  
 ),  
 Column(  
 children: [  
 InkWell(  
 onTap: () {  
 Get.to(() => paymentMethodList[  
 selectedPaymentMethod  
 .value] ==  
 "Credit/Debit Card"  
 ? AddCard(  
 selectedMethod:  
 paymentMethodList[  
 selectedPaymentMethod  
 .value],  
 )  
 : SelectBankScreen(  
 selectedMethod:  
 paymentMethodList[  
 selectedPaymentMethod  
 .value],  
 ));  
 },  
 child: fillColorButton(  
 text: 'Add'.tr,  
 color: pinkAppColor),  
 ),  
 ],  
 ),  
 ],  
 ),  
 );  
 },  
 );  
 },  
 child: fillColorButton(  
 text: 'Add New'.tr, color: pinkAppColor)),  
 getVerSpace(4.h),  
 ],  
 ),  
 ),  
 ),  
 )  
 ],  
 ),  
 ),  
 );  
 }  
}

# SettingScreens

## notification\_screen

import 'package:flutter/material.dart';  
import 'package:get/get.dart';  
import 'package:responsive\_sizer/responsive\_sizer.dart';  
  
import '../../utils/color\_data.dart';  
import '../Widgets/widget\_utils.dart';  
  
class NotificationScreen extends StatefulWidget {  
 const NotificationScreen({Key? key}) : super(key: key);  
  
 @override  
 State<NotificationScreen> createState() => \_NotificationScreenState();  
}  
  
class \_NotificationScreenState extends State<NotificationScreen> {  
 var selectedPaymentMethod = 0.obs;  
 back(){  
 Get.back();  
 }  
  
 @override  
 Widget build(BuildContext context) {  
 return WillPopScope(  
 onWillPop: () {  
 return back();  
 },  
 child: Scaffold(  
 backgroundColor: backgroundColor,  
 body: SizedBox(  
 width: 100.w,  
 child: Column(  
 children: [  
 SizedBox(  
 width: 100.h,  
 child: Stack(  
 children: [  
 Container(  
 height: 20.h,  
 width: 100.w,  
 alignment: Alignment.center,  
 padding: EdgeInsets.only(left: 3.h, right: 3.h),  
 decoration: BoxDecoration(  
 color: accentColor,  
 borderRadius: BorderRadius.only(  
 bottomRight: Radius.circular(4.h))),  
 child: Row(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 mainAxisAlignment: MainAxisAlignment.start,  
 children: [  
 GestureDetector(  
 onTap: () {  
 Get.back();  
 },  
 child: Icon(Icons.arrow\_back\_ios\_new\_outlined,  
 color: backgroundColor, size: 2.5.h),  
 ),  
 const Spacer(),  
 getCustomTextW6S22(  
 text: 'Notifications'.tr,  
 color: textWhiteColor,  
 ),  
 const Spacer()  
 ],  
 ),  
 ),  
 Positioned(  
 top: 15.h,  
 right: 0,  
 left: 0,  
 child: Container(  
 width: 100.h,  
 height: 5.h,  
 padding:  
 EdgeInsets.symmetric(vertical: 2.h, horizontal: 3.h),  
 alignment: Alignment.topCenter,  
 margin: EdgeInsets.symmetric(horizontal: 3.h),  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius: BorderRadius.only(  
 topRight: Radius.circular(10.px),  
 topLeft: Radius.circular(10.px)),  
 ),  
 ),  
 ),  
 ],  
 ),  
 ),  
 getVerSpace(1.h),  
 Expanded(  
 child: SizedBox(  
 width: 100.w,  
 child: Padding(  
 padding: EdgeInsets.symmetric(horizontal: 4.0.h),  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.start,  
 children: [  
 getCustomTextW6S16(  
 text: 'Today\'s Notification'.tr, color: accentColor),  
 getVerSpace(2.h),  
 Expanded(  
 child: SizedBox(  
 child: ListView.builder(  
 padding: EdgeInsets.zero,  
 itemCount: 3,  
 shrinkWrap: true,  
 physics: const BouncingScrollPhysics(),  
 itemBuilder: (context, index) {  
 return Container(  
 width: 100.w,  
 padding: EdgeInsets.symmetric(  
 horizontal: 3.h, vertical: 1.h),  
 margin: EdgeInsets.symmetric(vertical: 0.5.h,horizontal: 1.h),  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius:  
 BorderRadius.all(Radius.circular(10.px)),  
 boxShadow: [  
 BoxShadow(  
 color: borderColor,  
 blurRadius: 2,  
 offset: const Offset(  
 0, 0), // changes position of shadow  
 ),  
 ],  
 ),  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.start,  
 mainAxisAlignment: MainAxisAlignment.center,  
 children: [  
 getCustomTextW6S15(  
 text: 'Approved Review'.tr,  
 color: accentColor),  
 getVerSpace(0.5.h),  
 getCustomTextW6S12(  
 text: 'Your Given Review about Shop is approved from Admin.'.tr,  
 color: subTextColor),  
 ],  
 ),  
 );  
 },  
 ),  
 ),  
 ),  
 getVerSpace(4.h),  
 ],  
 ),  
 ),  
 ),  
 )  
 ],  
 ),  
 ),  
 ),  
 );  
 }  
}

# SettingScreens

## personal\_information\_screen

import 'package:flutter/material.dart';  
import 'package:get/get.dart';  
import 'package:responsive\_sizer/responsive\_sizer.dart';  
  
import '../../utils/color\_data.dart';  
import '../Widgets/widget\_utils.dart';  
  
class PersonalInformationScreen extends StatefulWidget {  
 const PersonalInformationScreen({Key? key}) : super(key: key);  
  
 @override  
 State<PersonalInformationScreen> createState() =>  
 \_PersonalInformationScreenState();  
}  
  
class \_PersonalInformationScreenState extends State<PersonalInformationScreen> {  
 @override  
 Widget build(BuildContext context) {  
 return Scaffold(  
 resizeToAvoidBottomInset: false,  
 backgroundColor: backgroundColor,  
 body: SizedBox(  
 width: 100.w,  
 child: Column(  
 children: [  
 SizedBox(  
 width: 100.h,  
 child: Stack(  
 children: [  
 Container(  
 height: 20.h,  
 width: 100.w,  
 alignment: Alignment.center,  
 padding: EdgeInsets.only(left: 3.h, right: 3.h),  
 decoration: BoxDecoration(  
 color: accentColor,  
 borderRadius: BorderRadius.only(  
 bottomRight: Radius.circular(4.h))),  
 child: Row(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 mainAxisAlignment: MainAxisAlignment.start,  
 children: [  
 GestureDetector(  
 onTap: () {  
 Get.back();  
 },  
 child: Icon(Icons.arrow\_back\_ios\_new\_outlined,  
 color: backgroundColor, size: 2.5.h),  
 ),  
 const Spacer(),  
 getCustomTextW6S22(  
 text: 'Profile'.tr,  
 color: textWhiteColor,  
 ),  
 const Spacer()  
 ],  
 ),  
 ),  
 Positioned(  
 top: 15.h,  
 right: 0,  
 left: 0,  
 child: Container(  
 width: 100.h,  
 height: 5.h,  
 padding:  
 EdgeInsets.symmetric(vertical: 2.h, horizontal: 3.h),  
 alignment: Alignment.topCenter,  
 margin: EdgeInsets.symmetric(horizontal: 3.h),  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius: BorderRadius.only(  
 topRight: Radius.circular(10.px),  
 topLeft: Radius.circular(10.px)),  
 ),  
 ),  
 ),  
 ],  
 ),  
 ),  
 getVerSpace(1.h),  
 Expanded(  
 child: SizedBox(  
 width: 100.w,  
 child: SingleChildScrollView(  
 child: Padding(  
 padding: EdgeInsets.symmetric(horizontal: 4.0.h),  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.start,  
 children: [  
 getCustomTextW6S16(  
 text: 'Personal Information'.tr,  
 color: accentColor),  
 getVerSpace(2.h),  
 personalInfoCard(  
 title: 'Full Name'.tr,  
 subTitle: 'Muhammad Usman'.tr,  
 ),  
 getVerSpace(1.5.h),  
 personalInfoCard(  
 title: 'ID Number'.tr,  
 subTitle: '000-0000000-00-0'.tr,  
 ),  
 getVerSpace(1.5.h),  
 personalInfoCard(  
 title: 'Nationality'.tr,  
 subTitle: 'United Arab Emirates'.tr,  
 ),  
 getVerSpace(1.5.h),  
 personalInfoCard(  
 title: 'Gender'.tr,  
 subTitle: 'Male'.tr,  
 ),  
 getVerSpace(1.5.h),  
 personalInfoCard(  
 title: 'Card Number'.tr,  
 subTitle: '1234567890'.tr,  
 ),  
 getVerSpace(1.5.h),  
 personalInfoCard(  
 title: 'Date Of Birth'.tr,  
 subTitle: '12-03-1996'.tr,  
 ),  
 getVerSpace(1.5.h),  
 personalInfoCard(  
 title: 'Expiry Date'.tr,  
 subTitle: '12-03-2025'.tr,  
 ),  
 getVerSpace(1.5.h),  
 getVerSpace(4.h),  
 ],  
 ),  
 ),  
 ),  
 ),  
 )  
 ],  
 ),  
 ),  
 );  
 }  
}

# SettingScreens

## setting\_screen

import 'package:algorthimi/view/SettingScreens/notification\_screen.dart';  
import 'package:algorthimi/view/SettingScreens/personal\_information\_screen.dart';  
import 'package:flutter/material.dart';  
import 'package:get/get.dart';  
import 'package:responsive\_sizer/responsive\_sizer.dart';  
  
import '../../utils/color\_data.dart';  
import '../Widgets/widget\_utils.dart';  
import 'accounts\_screen.dart';  
  
class SettingScreen extends StatefulWidget {  
 const SettingScreen({Key? key}) : super(key: key);  
  
 @override  
 State<SettingScreen> createState() => \_SettingScreenState();  
}  
  
class \_SettingScreenState extends State<SettingScreen> {  
 var language = 'English'.obs;  
  
 @override  
 Widget build(BuildContext context) {  
 return Scaffold(  
 resizeToAvoidBottomInset: false,  
 backgroundColor: backgroundColor,  
 body: SizedBox(  
 width: 100.w,  
 child: Column(  
 children: [  
 SizedBox(  
 width: 100.h,  
 child: Stack(  
 children: [  
 Container(  
 height: 20.h,  
 width: 100.w,  
 alignment: Alignment.topCenter,  
 padding: EdgeInsets.only(left: 3.h, right: 3.h),  
 decoration: BoxDecoration(  
 color: accentColor,  
 borderRadius: BorderRadius.only(  
 bottomRight: Radius.circular(4.h))),  
 child: Row(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 mainAxisAlignment: MainAxisAlignment.start,  
 children: [  
 GestureDetector(  
 onTap: () {},  
 child: Container(),  
 ),  
 const Spacer(),  
 getCustomTextW6S22(  
 text: 'Settings'.tr,  
 color: textWhiteColor,  
 ),  
 const Spacer()  
 ],  
 ),  
 ),  
 Positioned(  
 top: 15.h,  
 right: 0,  
 left: 0,  
 child: Container(  
 width: 100.h,  
 height: 5.h,  
 padding:  
 EdgeInsets.symmetric(vertical: 2.h, horizontal: 3.h),  
 alignment: Alignment.topCenter,  
 margin: EdgeInsets.symmetric(horizontal: 3.h),  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius: BorderRadius.only(  
 topRight: Radius.circular(10.px),  
 topLeft: Radius.circular(10.px)),  
 ),  
 ),  
 ),  
 ],  
 ),  
 ),  
 getVerSpace(1.h),  
 Expanded(  
 child: SizedBox(  
 width: 100.w,  
 child: SingleChildScrollView(  
 child: Padding(  
 padding: EdgeInsets.symmetric(horizontal: 4.0.h),  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.start,  
 children: [  
 getCustomTextW6S16(  
 text: 'Your Account'.tr, color: accentColor),  
 getVerSpace(2.h),  
 GestureDetector(  
 onTap: () {  
 Get.to(() => const PersonalInformationScreen());  
 },  
 child: settingFirstCard(  
 title: 'Personal Information'.tr,  
 subTitle: 'Muhammad Usman'.tr,  
 arrowColor: borderColor,  
 icon: 'profile\_icon.svg'),  
 ),  
 getVerSpace(1.5.h),  
 GestureDetector(  
 onTap: () {  
 Get.to(() => const AccountsScreen());  
 },  
 child: settingCard(  
 icon: 'card\_icon.svg',  
 arrowColor: borderColor,  
 title: 'Accounts'.tr),  
 ),  
 getVerSpace(1.5.h),  
 GestureDetector(  
 onTap: () {  
 Get.to(() => const NotificationScreen());  
 },  
 child: settingCard(  
 icon: 'notification\_icon.svg',  
 arrowColor: borderColor,  
 title: 'Notifications'.tr),  
 ),  
 getVerSpace(1.5.h),  
 getCustomTextW6S16(  
 text: 'Benefits'.tr, color: accentColor),  
 getVerSpace(2.h),  
 settingFirstCard(  
 title: 'Rewards'.tr,  
 subTitle: 'You have 568 points'.tr,  
 arrowColor: borderColor,  
 icon: 'reward\_icon.svg'),  
 getVerSpace(1.5.h),  
 getCustomTextW6S16(  
 text: 'Support'.tr, color: accentColor),  
 getVerSpace(2.h),  
 settingCard(  
 icon: 'help\_icon.svg',  
 arrowColor: borderColor,  
 title: 'Help'.tr),  
 getVerSpace(1.5.h),  
 settingCard(  
 icon: 'rate\_icon.svg',  
 arrowColor: borderColor,  
 title: 'Rate the App'.tr),  
 getVerSpace(1.5.h),  
 getCustomTextW6S16(  
 text: 'Preferences'.tr, color: accentColor),  
 getVerSpace(2.h),  
 GestureDetector(  
 onTap: () {  
 showLanguageDialog();  
 },  
 child: Obx(  
 () => settingFirstCard(  
 title: 'Language'.tr,  
 subTitle: language.value.toString().tr,  
 arrowColor: borderColor,  
 icon: 'language\_icon.svg'),  
 ),  
 ),  
 getVerSpace(3.h),  
 signOutButton(),  
 getVerSpace(4.h),  
 ],  
 ),  
 ),  
 ),  
 ),  
 )  
 ],  
 ),  
 ),  
 );  
 }  
  
 Future<void> showLanguageDialog() async {  
 await showDialog<void>(  
 context: context,  
 builder: (BuildContext context) {  
 return SimpleDialog(  
 elevation: 0,  
 title: getCustomTextW6S20(  
 text: 'Select Language'.tr, color: accentColor),  
 children: <Widget>[  
 SimpleDialogOption(  
 onPressed: () {  
 Get.updateLocale(const Locale('en', 'US'));  
 language.value = 'English';  
 Get.back();  
 },  
 child:  
 getCustomTextW6S15(text: 'English', color: textBlackColor),  
 ),  
 SimpleDialogOption(  
 onPressed: () {  
 Get.updateLocale(const Locale('ar', 'SA'));  
 language.value = 'عربي';  
 Get.back();  
 },  
 child: getCustomTextW6S15(text: 'عربي', color: textBlackColor),  
 ),  
 SimpleDialogOption(  
 onPressed: () {  
 Get.updateLocale(const Locale('fr', 'FR'));  
 language.value = 'Français';  
  
 Get.back();  
 },  
 child:  
 getCustomTextW6S15(text: 'Français', color: textBlackColor),  
 ),  
 SimpleDialogOption(  
 onPressed: () {  
 Get.updateLocale(const Locale('de', 'DE'));  
 language.value = 'Deutsch';  
  
 Get.back();  
 },  
 child:  
 getCustomTextW6S15(text: 'Deutsch', color: textBlackColor),  
 ),  
 SimpleDialogOption(  
 onPressed: () {  
 Get.updateLocale(const Locale('ur', 'PK'));  
 language.value = 'اردو';  
  
 Get.back();  
 },  
 child: getCustomTextW6S15(text: 'اردو', color: textBlackColor),  
 ),  
 SimpleDialogOption(  
 onPressed: () {  
 Get.updateLocale(const Locale('hi', 'IN'));  
 language.value = 'हिंदी';  
  
 Get.back();  
 },  
 child: getCustomTextW6S15(text: 'हिंदी', color: textBlackColor),  
 ),  
 SimpleDialogOption(  
 onPressed: () {  
 Get.updateLocale(const Locale('ml', 'IN'));  
 language.value = 'മലയാളം';  
  
 Get.back();  
 },  
 child:  
 getCustomTextW6S15(text: 'മലയാളം', color: textBlackColor),  
 ),  
 SimpleDialogOption(  
 onPressed: () {  
 Get.updateLocale(const Locale('bn', 'BD'));  
 language.value = 'বাংলা';  
  
 Get.back();  
 },  
 child: getCustomTextW6S15(text: 'বাংলা', color: textBlackColor),  
 ),  
 SimpleDialogOption(  
 onPressed: () {  
 Get.updateLocale(const Locale('ne', 'IN'));  
 language.value = 'नेपाली';  
  
 Get.back();  
 },  
 child:  
 getCustomTextW6S15(text: 'नेपाली', color: textBlackColor),  
 ),  
 ],  
 );  
 });  
 }  
}

# SplashLanguageScreens

## language\_screen

import 'dart:async';  
  
import 'package:algorthimi/utils/color\_data.dart';  
import 'package:algorthimi/utils/data.dart';  
import 'package:flutter/material.dart';  
import 'package:get/get.dart';  
import 'package:responsive\_sizer/responsive\_sizer.dart';  
  
import '../AuthScreens/login\_screen.dart';  
import '../Widgets/widget\_utils.dart';  
  
var language = 'Select your preferred language'.obs;  
  
class LanguageScreen extends StatefulWidget {  
 const LanguageScreen({Key? key}) : super(key: key);  
  
 @override  
 State<LanguageScreen> createState() => \_LanguageScreenState();  
}  
  
class \_LanguageScreenState extends State<LanguageScreen> {  
 var isOpenLanguage = false.obs;  
 var selectedLanguage = 10001.obs;  
  
 @override  
 Widget build(BuildContext context) {  
 return Obx(  
 () => Scaffold(  
 backgroundColor: accentColor,  
 body: Container(  
 height: 100.h,  
 width: 100.h,  
 margin: EdgeInsets.symmetric(horizontal: 3.h, vertical: 8.h),  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 mainAxisAlignment: MainAxisAlignment.start,  
 children: [  
 getVerSpace(8.h),  
 Align(  
 alignment: Alignment.center,  
 child: getSvgImage('app\_logo.svg', height: 12.h)),  
 getVerSpace(1.h),  
 getCustomTextW6S22(text: 'ALGORITHMI', color: textWhiteColor),  
 getVerSpace(5.h),  
 GestureDetector(  
 onTap: () {  
 isOpenLanguage.value = !isOpenLanguage.value;  
 },  
 child: Container(  
 padding: EdgeInsets.symmetric(horizontal: 2.h, vertical: 1.h),  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 border: Border.all(color: backgroundColor, width: 1.px),  
 borderRadius: BorderRadius.all(Radius.circular(10.px))),  
 child: Row(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 mainAxisAlignment: MainAxisAlignment.spaceBetween,  
 children: [  
 getCustomTextW6S15(  
 text: language.toString().tr,  
 color: textBlackColor,  
 ),  
 Icon(  
 Icons.arrow\_drop\_down\_rounded,  
 color: textBlackColor,  
 )  
 ],  
 ),  
 ),  
 ),  
 getVerSpace(1.h),  
 isOpenLanguage.value  
 ? Flexible(  
 child: Container(  
 padding: EdgeInsets.symmetric(  
 horizontal: 2.5.h, vertical: 1.h),  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 border: Border.all(  
 color: backgroundColor, width: 1.px),  
 borderRadius:  
 BorderRadius.all(Radius.circular(10.px))),  
 child: ListView.builder(  
 shrinkWrap: true,  
 physics: const NeverScrollableScrollPhysics(),  
 itemCount: DataFile.languageList.length,  
 padding: EdgeInsets.zero,  
 itemBuilder: (context, index) {  
 return GestureDetector(  
 onTap: () {  
 selectedLanguage.value = index;  
  
 if (DataFile.languageList[index].name  
 .toString() ==  
 'English') {  
 Get.updateLocale(const Locale('en', 'US'));  
 } else if (DataFile.languageList[index].name  
 .toString() ==  
 'اردو') {  
 Get.updateLocale(const Locale('ur', 'PK'));  
 } else if (DataFile.languageList[index].name  
 .toString() ==  
 'عربي') {  
 Get.updateLocale(const Locale('ar', 'SA'));  
 } else if (DataFile.languageList[index].name  
 .toString() ==  
 'Français') {  
 Get.updateLocale(const Locale('fr', 'FR'));  
 } else if (DataFile.languageList[index].name  
 .toString() ==  
 'Deutsch') {  
 Get.updateLocale(const Locale('de', 'DE'));  
 } else if (DataFile.languageList[index].name  
 .toString() ==  
 'हिंदी') {  
 Get.updateLocale(const Locale('hi', 'IN'));  
 } else if (DataFile.languageList[index].name  
 .toString() ==  
 'മലയാളം') {  
 Get.updateLocale(const Locale('ml', 'IN'));  
 } else if (DataFile.languageList[index].name  
 .toString() ==  
 'नेपाली') {  
 Get.updateLocale(const Locale('ne', 'IN'));  
 } else if (DataFile.languageList[index].name  
 .toString() ==  
 'বাংলা') {  
 Get.updateLocale(const Locale('bn', 'BD'));  
 }  
  
 language.value = DataFile  
 .languageList[index].name  
 .toString();  
  
 Timer(const Duration(milliseconds: 100), () {  
 Get.to(() => const LoginScreen());  
 });  
 },  
 child: Container(  
 color: backgroundColor,  
 margin: EdgeInsets.symmetric(vertical: 1.h),  
 child: Row(  
 crossAxisAlignment:  
 CrossAxisAlignment.center,  
 mainAxisAlignment: MainAxisAlignment.start,  
 children: [  
 Container(  
 width: 1.8.h,  
 height: 1.8.h,  
 decoration: BoxDecoration(  
 shape: BoxShape.circle,  
 border: Border.all(  
 color: selectedLanguage.value ==  
 index  
 ? pinkAppColor  
 : borderColor,  
 width: selectedLanguage.value ==  
 index  
 ? 4.px  
 : 1.5.px)),  
 ),  
 getHorSpace(1.2.h),  
 getCustomTextW6S15(  
 text: DataFile.languageList[index].name  
 .toString()  
 .tr,  
 color: textBlackColor,  
 ),  
 ],  
 ),  
 ),  
 );  
 },  
 )),  
 )  
 : Container()  
 ],  
 ),  
 ),  
 ),  
 );  
 }  
}

# SplashLanguageScreens

## splash\_screen

import 'dart:async';  
  
import 'package:flutter/material.dart';  
import 'package:get/get.dart';  
import 'package:responsive\_sizer/responsive\_sizer.dart';  
  
import '../../utils/color\_data.dart';  
import '../Widgets/widget\_utils.dart';  
import 'language\_screen.dart';  
  
class SplashScreen extends StatefulWidget {  
 const SplashScreen({Key? key}) : super(key: key);  
  
 @override  
 State<SplashScreen> createState() => \_SplashScreenState();  
}  
  
class \_SplashScreenState extends State<SplashScreen> {  
 @override  
 void initState() {  
 super.initState();  
 \_getIsFirst();  
 }  
  
 \_getIsFirst() async {  
 // bool isIntro = await PrefData.getIsIntro();  
 // bool isSignIn = await PrefData.getIsSignIn();  
 //  
 // if (isIntro) {  
 // Constant.sendToNext(context, Routes.introScreenRoute);  
 // } else if (!isSignIn) {  
 // Constant.sendToNext(context, Routes.loginScreenRoute);  
 // } else {  
 // Timer(const Duration(seconds: 3), () {  
 // Constant.sendToNext(context, Routes.homeScreenRoute);  
 // });  
 // }  
 Timer(const Duration(seconds: 3), () {  
 Get.offAll(()=> const LanguageScreen());  
 });  
 }  
  
 @override  
 Widget build(BuildContext context) {  
 return Scaffold(  
 body: Container(  
 height: 100.h,  
 width: 100.h,  
 color: accentColor,  
 child: Center(  
 child: getSvgImage('splash.svg', width: 26.2.h, height: 19.7.h)),  
 ),  
 );  
 }  
}

# WalletScreens

## add\_bank

import 'package:flutter/material.dart';  
import 'package:get/get.dart';  
import 'package:responsive\_sizer/responsive\_sizer.dart';  
  
import '../../utils/color\_data.dart';  
import '../Widgets/widget\_utils.dart';  
import 'opt\_screen.dart';  
  
//ignore: must\_be\_immutable  
class AddBankScreen extends StatefulWidget {  
 AddBankScreen({Key? key, this.selectedMethod, this.selectedBank})  
 : super(key: key);  
 dynamic selectedMethod;  
 dynamic selectedBank;  
  
 @override  
 State<AddBankScreen> createState() => \_AddBankScreenState();  
}  
  
class \_AddBankScreenState extends State<AddBankScreen> {  
 TextEditingController accountNumberController = TextEditingController();  
  
 @override  
 Widget build(BuildContext context) {  
 return Scaffold(  
 body: SizedBox(  
 width: 100.h,  
 child: Column(  
 children: [  
 Container(  
 height: 20.h,  
 padding: EdgeInsets.symmetric(horizontal: 3.h, vertical: 2.h),  
 decoration: BoxDecoration(color: accentColor),  
 child: Column(  
 mainAxisAlignment: MainAxisAlignment.spaceBetween,  
 crossAxisAlignment: CrossAxisAlignment.center,  
 children: [  
 getVerSpace(2.h),  
 Row(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 mainAxisAlignment: MainAxisAlignment.start,  
 children: [  
 GestureDetector(  
 onTap: () {  
 Get.back();  
 },  
 child: Icon(Icons.arrow\_back\_ios\_new\_outlined,  
 color: backgroundColor, size: 2.5.h),  
 ),  
 const Spacer(),  
 getCustomTextW6S22(  
 text: widget.selectedMethod.toString(),  
 color: textWhiteColor,  
 ),  
 const Spacer()  
 ],  
 ),  
  
 Container(  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius: BorderRadius.all(Radius.circular(15.px)),  
 ),  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 mainAxisAlignment: MainAxisAlignment.spaceBetween,  
 children: [  
 ListTile(  
 contentPadding: EdgeInsets.symmetric(horizontal: 2.h),  
 leading: CircleAvatar(  
 radius: 2.5.h, backgroundColor: borderColor),  
 title: getCustomTextW6S15(  
 text: widget.selectedBank.toString().tr,  
 color: textBlackColor),  
 ),  
 ],  
 ),  
 ),  
 ],  
 ),  
 ),  
 Padding(  
 padding: EdgeInsets.symmetric(horizontal: 3.h),  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.start,  
 children: [  
 getVerSpace(3.h),  
 getCustomTextW6S15(  
 text: widget.selectedMethod == 'E-Wallet'.tr  
 ? 'Account Number'.tr  
 : 'Account Number/IBAN'.tr,  
 color: accentColor),  
 getVerSpace(1.h),  
 Container(  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius: BorderRadius.all(Radius.circular(15.px)),  
 boxShadow: [  
 BoxShadow(  
 color: borderColor,  
 blurRadius: 10,  
 offset: const Offset(  
 0, 0), // changes position of shadow  
 ),  
 ]),  
 child: getCustomTextFormField(  
 keyboardType: TextInputType.text,  
 obscureText: false,  
 hintText: widget.selectedMethod == 'E-Wallet'.tr  
 ? 'Account Number'.tr  
 : 'Account Number/IBAN'.tr,  
 controller: accountNumberController,  
 onChanged: (value) {},  
 ),  
 ),  
 getVerSpace(0.5.h),  
 getCustomTextW6S10(  
 text: widget.selectedMethod == 'E-Wallet'.tr  
 ? '${'Please enter'.tr} ${widget.selectedBank.toString().tr} ${'11 digit account number'.tr}'  
 : '${'Please enter'.tr} ${widget.selectedBank.toString().tr} ${'11 digit account number'.tr}',  
 color: subTextColor),  
 getVerSpace(5.h),  
 GestureDetector(  
 onTap: () {  
 Get.to(() => OTPScreen(  
 selectedMethod: widget.selectedMethod,  
 ));  
 },  
 child:  
 fillColorButton(text: 'Add'.tr, color: pinkAppColor)),  
 getVerSpace(1.5.h),  
 GestureDetector(  
 onTap: () {  
 Get.back();  
 Get.back();  
 Get.back();  
 },  
 child: outlineButton(  
 text: 'Cancel'.tr,  
 color: backgroundColor,  
 borderColor: pinkAppColor))  
 ],  
 ),  
 )  
 ],  
 ),  
 ),  
 );  
 }  
}

# WalletScreens

## add\_card

import 'package:flutter/material.dart';  
import 'package:get/get.dart';  
import 'package:responsive\_sizer/responsive\_sizer.dart';  
  
import '../../utils/color\_data.dart';  
import '../Widgets/widget\_utils.dart';  
  
//ignore: must\_be\_immutable  
class AddCard extends StatefulWidget {  
 AddCard({Key? key, this.selectedMethod}) : super(key: key);  
 dynamic selectedMethod;  
  
 @override  
 State<AddCard> createState() => \_AddCardState();  
}  
  
class \_AddCardState extends State<AddCard> {  
 TextEditingController cardNumberController = TextEditingController();  
 TextEditingController dateNumberController = TextEditingController();  
 TextEditingController ccvNumberController = TextEditingController();  
 TextEditingController nameNumberController = TextEditingController();  
  
 back() {  
 Get.back();  
 Get.back();  
 }  
  
 @override  
 Widget build(BuildContext context) {  
 return WillPopScope(  
 onWillPop: () {  
 return back();  
 },  
 child: Scaffold(  
 resizeToAvoidBottomInset: false,  
 body: SizedBox(  
 width: 100.h,  
 child: Column(  
 children: [  
 Container(  
 width: 100.w,  
 height: 15.h,  
 padding: EdgeInsets.symmetric(horizontal: 3.h, vertical: 2.h),  
 decoration: BoxDecoration(color: accentColor),  
 child: Column(  
 mainAxisAlignment: MainAxisAlignment.center,  
 crossAxisAlignment: CrossAxisAlignment.center,  
 children: [  
 getVerSpace(2.h),  
 getCustomTextW6S22(  
 text: widget.selectedMethod.toString(),  
 color: textWhiteColor,  
 ),  
 ],  
 ),  
 ),  
 getVerSpace(3.h),  
 Padding(  
 padding: EdgeInsets.symmetric(horizontal: 3.h),  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.start,  
 mainAxisAlignment: MainAxisAlignment.spaceBetween,  
 children: [  
 Column(  
 crossAxisAlignment: CrossAxisAlignment.start,  
 children: [  
 getCustomTextW6S15(  
 text: 'Card Number'.tr, color: accentColor),  
 getVerSpace(1.h),  
 Container(  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius: BorderRadius.all(Radius.circular(15.px)),  
 boxShadow: [  
 BoxShadow(  
 color: borderColor,  
 blurRadius: 10,  
 offset: const Offset(  
 0, 0), // changes position of shadow  
 ),  
 ],),  
 child: getCustomTextFormField(  
 keyboardType: TextInputType.phone,  
 obscureText: false,  
 hintText: 'Enter 16 digit card no'.tr,  
 controller: cardNumberController,  
 onChanged: (value) {},  
 ),  
 ),  
 getVerSpace(2.h),  
 Row(  
 children: [  
 Expanded(  
 flex: 2,  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.start,  
 children: [  
 getCustomTextW6S15(  
 text: 'Expiry Date'.tr,  
 color: accentColor),  
 getVerSpace(1.h),  
 Container(  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius: BorderRadius.all(Radius.circular(15.px)),  
 boxShadow: [  
 BoxShadow(  
 color: borderColor,  
 blurRadius: 10,  
 offset: const Offset(  
 0, 0), // changes position of shadow  
 ),  
 ],),  
 child: getCustomTextFormField(  
 keyboardType: TextInputType.phone,  
 obscureText: false,  
 hintText: '00/00',  
 controller: dateNumberController,  
 onChanged: (value) {},  
 ),  
 ),  
 ],  
 ),  
 ),  
 getHorSpace(1.h),  
 Expanded(  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.start,  
 children: [  
 getCustomTextW6S15(  
 text: 'CCV'.tr, color: accentColor),  
 getVerSpace(1.h),  
 Container(  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius: BorderRadius.all(Radius.circular(15.px)),  
 boxShadow: [  
 BoxShadow(  
 color: borderColor,  
 blurRadius: 10,  
 offset: const Offset(  
 0, 0), // changes position of shadow  
 ),  
 ],),  
 child: getCustomTextFormField(  
 keyboardType: TextInputType.phone,  
 obscureText: false,  
 hintText: '000',  
 controller: ccvNumberController,  
 onChanged: (value) {},  
 ),  
 ),  
 ],  
 ),  
 ),  
 ],  
 ),  
 getVerSpace(2.h),  
 getCustomTextW6S15(  
 text: 'Client'.tr, color: accentColor),  
 getVerSpace(1.h),  
 Container(  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius: BorderRadius.all(Radius.circular(15.px)),  
 boxShadow: [  
 BoxShadow(  
 color: borderColor,  
 blurRadius: 10,  
 offset: const Offset(  
 0, 0), // changes position of shadow  
 ),  
 ],),  
 child: getCustomTextFormField(  
 keyboardType: TextInputType.text,  
 obscureText: false,  
 hintText: 'Enter Your Name'.tr,  
 controller: nameNumberController,  
 onChanged: (value) {},  
 ),  
 ),  
 ],  
 ),  
 getVerSpace(5.h),  
 Column(  
 children: [  
 GestureDetector(  
 onTap: () {  
 // Get.to(() => OTPScreen(  
 // isAddBankConfirm: true,  
 // selectedMethod: widget.selectedMethod,  
 // ));  
 },  
 child: fillColorButton(  
 text: 'Add'.tr, color: pinkAppColor)),  
 getVerSpace(1.5.h),  
 GestureDetector(  
 onTap: () {  
 Get.back();  
 Get.back();  
 },  
 child: outlineButton(  
 text: 'Cancel'.tr,  
 color: backgroundColor,  
 borderColor: pinkAppColor))  
 ],  
 )  
 ],  
 ),  
 ),  
 ],  
 ),  
 ),  
 ),  
 );  
 }  
}

# WalletScreens

## al\_ansari\_screen

import 'package:algorthimi/view/BottomNavBar/bottom\_nav\_bar\_freelancer\_screen.dart';  
import 'package:flutter/material.dart';  
import 'package:gallery\_saver/gallery\_saver.dart';  
import 'package:get/get.dart';  
import 'package:responsive\_sizer/responsive\_sizer.dart';  
  
import '../../utils/color\_data.dart';  
import '../Widgets/widget\_utils.dart';  
  
import 'dart:io';  
import 'package:flutter\_native\_screenshot/flutter\_native\_screenshot.dart';  
import 'package:http/http.dart' as http;  
  
class AlAnsariExchangeScreen extends StatefulWidget {  
 const AlAnsariExchangeScreen({Key? key}) : super(key: key);  
  
 @override  
 State<AlAnsariExchangeScreen> createState() => \_AlAnsariExchangeScreenState();  
}  
  
class \_AlAnsariExchangeScreenState extends State<AlAnsariExchangeScreen> {  
  
  
 Future<void> \_takeScreenshot(BuildContext context) async {  
 try {  
 final screenshotPath = await FlutterNativeScreenshot.takeScreenshot();  
 await GallerySaver.saveImage(screenshotPath!);  
 ScaffoldMessenger.of(context).showSnackBar(  
 SnackBar(  
 content: Text('Your details saved to gallery'),  
 ),  
 );  
 } catch (e) {  
 print("Error saving screenshot: $e");  
 ScaffoldMessenger.of(context).showSnackBar(  
 SnackBar(  
 content: Text('Failed to save your detail'),  
 ),  
 );  
 }  
 }  
  
  
  
 ////////////////transfer data for Api//////////////  
  
  
  
  
 // Future<void> \_takeScreenshot(BuildContext context) async {  
 // try {  
 // final screenshotPath = await FlutterNativeScreenshot.takeScreenshot();  
 //  
 // // Send the screenshot to your server for S3 upload  
 // var request = http.MultipartRequest(  
 // 'POST', Uri.parse('https://algo-app-api.ytech.systems/upload'));  
 // request.fields['image'] = 'image'; // Set API key as a form field with the name 'image'  
 // request.files.add(await http.MultipartFile.fromPath('image', screenshotPath!));  
 //  
 // var response = await request.send();  
 //  
 // if (response.statusCode == 200) {  
 // ScaffoldMessenger.of(context).showSnackBar(  
 // SnackBar(  
 // content: Text('Screenshot uploaded to S3 successfully'),  
 // ),  
 // );  
 // } else {  
 // ScaffoldMessenger.of(context).showSnackBar(  
 // SnackBar(  
 // content: Text('Failed to upload screenshot to S3'),  
 // ),  
 // );  
 // }  
 // } catch (e) {  
 // print("Error saving screenshot: $e");  
 // ScaffoldMessenger.of(context).showSnackBar(  
 // SnackBar(  
 // content: Text('Failed to save screenshot'),  
 // ),  
 // );  
 // }  
 // }  
  
  
 @override  
 Widget build(BuildContext context) {  
 return Scaffold(  
 resizeToAvoidBottomInset: false,  
 backgroundColor: backgroundColor,  
 body: SizedBox(  
 height: 100.h,  
 width: 100.h,  
 child: Stack(  
 children: [  
 Container(  
 height: 20.h,  
 width: 100.w,  
 alignment: Alignment.topCenter,  
 padding: EdgeInsets.only(top: 6.h, left: 3.h, right: 3.h),  
 decoration: BoxDecoration(  
 color: accentColor,  
 borderRadius:  
 BorderRadius.only(bottomRight: Radius.circular(4.h))),  
 child: Row(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 mainAxisAlignment: MainAxisAlignment.start,  
 children: [  
 GestureDetector(  
 onTap: () {  
 Get.back();  
 },  
 child: Icon(Icons.arrow\_back\_ios\_new\_outlined,  
 color: backgroundColor, size: 2.5.h),  
 ),  
 const Spacer(),  
 getCustomTextW6S22(  
 text: 'Congratulations',  
 color: textWhiteColor,  
 ),  
 const Spacer()  
 ],  
 ),  
 ),  
 Positioned(  
 top: 15.h,  
 right: 0,  
 left: 0,  
 child: Container(  
 width: 100.h,  
 height: 100.h - 20.h,  
 padding: EdgeInsets.only(top: 2.h),  
 alignment: Alignment.topCenter,  
 margin: EdgeInsets.symmetric(horizontal: 3.h),  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius: BorderRadius.all(Radius.circular(10.px))),  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 mainAxisAlignment: MainAxisAlignment.spaceBetween,  
 children: [  
 getCustomTextW6S15(  
 text: 'Get cash from\nAl Ansari Exchange',  
 color: textBlackColor,  
 textAlign: TextAlign.center),  
 Stack(  
 alignment: Alignment.center,  
 children: [  
 getSvgImage('back\_logo.svg', height: 29.h),  
 getSvgImage('qr\_code.svg', height: 22.5.h),  
 ],  
 ),  
 Column(  
 children: [  
 GestureDetector(  
 onTap: () =>\_takeScreenshot(context),  
 child: fillColorButton(  
 color: pinkAppColor, text: 'Download Receipt'),  
 ),  
 getVerSpace(1.5.h),  
 GestureDetector(  
 onTap: () {},  
 child: outlineButton(  
 color: backgroundColor,  
 borderColor: pinkAppColor,  
 text: 'Take Screenshot'),  
 ),  
 getVerSpace(1.5.h),  
 GestureDetector(  
 onTap: () {  
 Get.offAll(() => const BottomNavBarFreelancerScreen());  
 },  
 child: outlineButton(  
 color: backgroundColor,  
 borderColor: pinkAppColor,  
 text: 'Go to Dashboard'),  
 )  
 ],  
 ),  
 ],  
 ),  
 ),  
 ),  
 ],  
 ),  
 ),  
 );  
 }  
}

# WalletScreens

## congratulation\_screen

import 'package:algorthimi/view/BottomNavBar/bottom\_nav\_bar\_freelancer\_screen.dart';  
import 'package:flutter/material.dart';  
import 'package:gallery\_saver/gallery\_saver.dart';  
import 'package:get/get.dart';  
import 'package:responsive\_sizer/responsive\_sizer.dart';  
  
import '../../utils/color\_data.dart';  
import '../Widgets/widget\_utils.dart';  
  
import 'dart:io';  
import 'package:flutter\_native\_screenshot/flutter\_native\_screenshot.dart';  
import 'package:http/http.dart' as http;  
  
  
//ignore: must\_be\_immutable  
class CongratulationTransferBankEWalletScreen extends StatefulWidget {  
 CongratulationTransferBankEWalletScreen({Key? key, this.selectedMethod})  
 : super(key: key);  
 dynamic selectedMethod;  
  
 @override  
 State<CongratulationTransferBankEWalletScreen> createState() =>  
 \_CongratulationTransferBankEWalletScreenState();  
}  
  
class \_CongratulationTransferBankEWalletScreenState  
 extends State<CongratulationTransferBankEWalletScreen> {  
  
  
 Future<void> \_takeScreenshot(BuildContext context) async {  
 try {  
 final screenshotPath = await FlutterNativeScreenshot.takeScreenshot();  
 await GallerySaver.saveImage(screenshotPath!);  
 ScaffoldMessenger.of(context).showSnackBar(  
 SnackBar(  
 content: Text('Your details saved to gallery'),  
 ),  
 );  
 } catch (e) {  
 print("Error saving screenshot: $e");  
 ScaffoldMessenger.of(context).showSnackBar(  
 SnackBar(  
 content: Text('Failed to save your detail'),  
 ),  
 );  
 }  
 }  
  
  
  
 ////////////////transfer data for Api//////////////  
  
  
  
 // Future<void> \_takeScreenshot(BuildContext context) async {  
 // try {  
 // final screenshotPath = await FlutterNativeScreenshot.takeScreenshot();  
 //  
 // // Send the screenshot to your server for S3 upload  
 // var request = http.MultipartRequest(  
 // 'POST', Uri.parse('https://algo-app-api.ytech.systems/upload'));  
 // request.fields['image'] = 'image'; // Set API key as a form field with the name 'image'  
 // request.files.add(await http.MultipartFile.fromPath('image', screenshotPath!));  
 //  
 // var response = await request.send();  
 //  
 // if (response.statusCode == 200) {  
 // ScaffoldMessenger.of(context).showSnackBar(  
 // SnackBar(  
 // content: Text('Screenshot uploaded to S3 successfully'),  
 // ),  
 // );  
 // } else {  
 // ScaffoldMessenger.of(context).showSnackBar(  
 // SnackBar(  
 // content: Text('Failed to upload screenshot to S3'),  
 // ),  
 // );  
 // }  
 // } catch (e) {  
 // print("Error saving screenshot: $e");  
 // ScaffoldMessenger.of(context).showSnackBar(  
 // SnackBar(  
 // content: Text('Failed to save screenshot'),  
 // ),  
 // );  
 // }  
 // }  
  
  
 back() {  
 Get.back();  
 Get.back();  
 }  
  
 @override  
 Widget build(BuildContext context) {  
 return WillPopScope(  
 onWillPop: () {  
 return back();  
 },  
 child: Scaffold(  
 resizeToAvoidBottomInset: false,  
 backgroundColor: backgroundColor,  
 body: SizedBox(  
 height: 100.h,  
 width: 100.h,  
 child: Stack(  
 children: [  
 Container(  
 height: 20.h,  
 width: 100.w,  
 alignment: Alignment.topCenter,  
 padding: EdgeInsets.only(top: 6.h, left: 3.h, right: 3.h),  
 decoration: BoxDecoration(  
 color: accentColor,  
 borderRadius:  
 BorderRadius.only(bottomRight: Radius.circular(4.h))),  
 child: Row(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 mainAxisAlignment: MainAxisAlignment.start,  
 children: [  
 GestureDetector(  
 onTap: () {  
 Get.back();  
 Get.back();  
 },  
 child: Icon(Icons.arrow\_back\_ios\_new\_outlined,  
 color: backgroundColor, size: 2.5.h),  
 ),  
 const Spacer(),  
 getCustomTextW6S22(  
 text: 'Congratulations',  
 color: textWhiteColor,  
 ),  
 const Spacer()  
 ],  
 ),  
 ),  
 Positioned(  
 top: 15.h,  
 right: 0,  
 left: 0,  
 child: Container(  
 width: 100.h,  
 height: 100.h - 20.h,  
 padding: EdgeInsets.only(top: 2.h),  
 alignment: Alignment.topCenter,  
 margin: EdgeInsets.symmetric(horizontal: 3.h),  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius: BorderRadius.all(Radius.circular(10.px))),  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 mainAxisAlignment: MainAxisAlignment.spaceBetween,  
 children: [  
 Column(  
 children: [  
 getSvgImage('congratulations\_smile.svg', height: 7.h),  
 getVerSpace(2.h),  
 widget.selectedMethod == 'withdrawCrypto'  
 ? Container()  
 : getCustomTextW6S20(  
 text: '500 AED', color: greenColor),  
 widget.selectedMethod == 'withdrawCrypto'  
 ? getCustomTextW6S15(  
 text:  
 'Your transaction has been\nSuccessful\n\n',  
 color: textBlackColor,  
 textAlign: TextAlign.center)  
 : getCustomTextW6S15(  
 text:  
 'has been successfully transferred to\nyour ${widget.selectedMethod} account\n\nRemaining Balance',  
 color: textBlackColor,  
 textAlign: TextAlign.center),  
 widget.selectedMethod == 'withdrawCrypto'  
 ? getCustomTextW6S20(  
 text: '9 Coins', color: accentColor)  
 : getCustomTextW6S20(  
 text: '500 AED', color: accentColor),  
 ],  
 ),  
 widget.selectedMethod == 'withdrawCrypto'  
 ? Stack(  
 alignment: Alignment.center,  
 children: [  
 getSvgImage('back\_logo.svg', height: 29.h),  
 getSvgImage('coin\_icon.svg', height: 14.h),  
 ],  
 )  
 : getSvgImage('back\_logo.svg', height: 29.h),  
 Column(  
 children: [  
 GestureDetector(  
 onTap: () =>\_takeScreenshot(context),  
 child: fillColorButton(  
 color: pinkAppColor,  
 text: widget.selectedMethod == 'withdrawCrypto'  
 ? 'Transaction History'  
 : 'Download Receipt'),  
 ),  
 getVerSpace(1.5.h),  
 GestureDetector(  
 onTap: () {  
 Get.offAll(() => const BottomNavBarFreelancerScreen());  
 },  
 child: outlineButton(  
 color: backgroundColor,  
 borderColor: pinkAppColor,  
 text: 'Go to Dashboard'),  
 )  
 ],  
 ),  
 ],  
 ),  
 ),  
 ),  
 ],  
 ),  
 ),  
 ),  
 );  
 }  
}

# WalletScreens

## opt\_screen

import 'dart:async';  
import 'dart:developer';  
  
import 'package:algorthimi/view/AuthScreens/welldone\_screen.dart';  
import 'package:flutter/material.dart';  
import 'package:get/get.dart';  
import 'package:pin\_code\_fields/pin\_code\_fields.dart';  
import 'package:responsive\_sizer/responsive\_sizer.dart';  
  
import '../../utils/color\_data.dart';  
import '../Widgets/widget\_utils.dart';  
  
//ignore: must\_be\_immutable  
class OTPScreen extends StatefulWidget {  
 OTPScreen({Key? key, this.selectedMethod}) : super(key: key);  
 dynamic selectedMethod;  
  
 @override  
 State<OTPScreen> createState() => \_OTPScreenState();  
}  
  
class \_OTPScreenState extends State<OTPScreen> {  
 TextEditingController otpController = TextEditingController();  
 StreamController<ErrorAnimationType>? errorController;  
  
 @override  
 Widget build(BuildContext context) {  
 return Scaffold(  
 resizeToAvoidBottomInset: false,  
 backgroundColor: backgroundColor,  
 body: SizedBox(  
 height: 100.h,  
 width: 100.h,  
 child: Stack(  
 children: [  
 Container(  
 height: 22.h,  
 width: 100.w,  
 alignment: Alignment.topCenter,  
 padding: EdgeInsets.only(top: 6.h, left: 3.h, right: 3.h),  
 decoration: BoxDecoration(  
 color: accentColor,  
 borderRadius:  
 BorderRadius.only(bottomRight: Radius.circular(4.h))),  
 child: Row(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 mainAxisAlignment: MainAxisAlignment.start,  
 children: [  
 GestureDetector(  
 onTap: () {  
 Get.back();  
 },  
 child: Icon(Icons.arrow\_back\_ios\_new\_outlined,  
 color: backgroundColor, size: 2.5.h),  
 ),  
 const Spacer(),  
 getCustomTextW6S22(  
 text: 'Enter Verification\nCode'.tr,  
 color: textWhiteColor,  
 ),  
 const Spacer()  
 ],  
 ),  
 ),  
 Positioned(  
 top: 16.h,  
 right: 0,  
 left: 0,  
 child: Container(  
 width: 100.h,  
 height: 100.h,  
 padding: EdgeInsets.only(top: 2.h),  
 alignment: Alignment.topCenter,  
 margin: EdgeInsets.symmetric(horizontal: 3.h),  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius: BorderRadius.all(Radius.circular(10.px))),  
 child: Column(  
 children: <Widget>[  
 Padding(  
 padding: EdgeInsets.symmetric(horizontal: 3.h),  
 child: getCustomTextW6S15(  
 text: 'Please enter the OTP Code\nwe sent you'.tr,  
 color: textBlackColor,  
 textAlign: TextAlign.center),  
 ),  
 getVerSpace(4.h),  
 PinCodeTextField(  
 length: 4,  
 keyboardType: TextInputType.phone,  
 appContext: context,  
 obscureText: false,  
 animationType: AnimationType.fade,  
 pinTheme: PinTheme(  
 borderWidth: 1.px,  
 shape: PinCodeFieldShape.box,  
 borderRadius: BorderRadius.circular(5),  
 selectedColor: pinkAppColor,  
 selectedFillColor: backgroundColor,  
 inactiveColor: subTextColor,  
 inactiveFillColor: backgroundColor,  
 fieldHeight: 5.5.h,  
 fieldWidth: 4.h,  
 activeColor: pinkAppColor,  
 activeFillColor: backgroundColor,  
 fieldOuterPadding:  
 EdgeInsets.symmetric(horizontal: 1.1.h)),  
 animationDuration: const Duration(milliseconds: 200),  
 backgroundColor: backgroundColor,  
 mainAxisAlignment: MainAxisAlignment.center,  
 enableActiveFill: true,  
 errorAnimationController: errorController,  
 controller: otpController,  
 onCompleted: (v) {  
 Get.to(() => WellDoneScreen(  
 screenTitle: 'addBank',  
 selectedMethod: widget.selectedMethod,  
 ));  
 log("Completed\n OTP:${otpController.text}");  
 },  
 onChanged: (value) {},  
 beforeTextPaste: (text) {  
 log("Allowing to paste $text");  
 return true;  
 },  
 ),  
 getVerSpace(2.h),  
 getCustomTextW6S15(  
 text: "I did not receive the OTP".tr,  
 color: subTextColor),  
 getVerSpace(1.8.h),  
 getCustomTextW6S15(  
 text: "Resend Code".tr, color: pinkAppColor),  
 getVerSpace(2.h),  
 ],  
 ),  
 ),  
 ),  
 ],  
 ),  
 ),  
 );  
 }  
}

# WalletScreens

## select\_bank

import 'dart:async';  
  
import 'package:flutter/material.dart';  
import 'package:get/get.dart';  
import 'package:responsive\_sizer/responsive\_sizer.dart';  
  
import '../../utils/color\_data.dart';  
import '../../utils/data.dart';  
import '../Widgets/widget\_utils.dart';  
import 'add\_bank.dart';  
  
//ignore: must\_be\_immutable  
class SelectBankScreen extends StatefulWidget {  
 SelectBankScreen({Key? key, this.selectedMethod}) : super(key: key);  
 dynamic selectedMethod;  
  
 @override  
 State<SelectBankScreen> createState() => \_SelectBankScreenState();  
}  
  
class \_SelectBankScreenState extends State<SelectBankScreen> {  
 TextEditingController searchController = TextEditingController();  
 var isClosed = true.obs;  
 var searchedList = [].obs;  
  
 back() {  
 Get.back();  
 Get.back();  
 }  
  
 @override  
 Widget build(BuildContext context) {  
 return WillPopScope(  
 onWillPop: () {  
 return back();  
 },  
 child: Obx(  
 () => Scaffold(  
 body: SizedBox(  
 width: 100.h,  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 children: [  
 Container(  
 height: 20.h,  
 padding: EdgeInsets.symmetric(horizontal: 3.h, vertical: 2.h),  
 decoration: BoxDecoration(color: accentColor),  
 child: Column(  
 mainAxisAlignment: MainAxisAlignment.spaceBetween,  
 crossAxisAlignment: CrossAxisAlignment.center,  
 children: [  
 getVerSpace(2.h),  
 Row(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 mainAxisAlignment: MainAxisAlignment.start,  
 children: [  
 GestureDetector(  
 onTap: () {  
 Get.back();  
 Get.back();  
 },  
 child: Icon(Icons.arrow\_back\_ios\_new\_outlined,  
 color: backgroundColor, size: 2.5.h),  
 ),  
 const Spacer(),  
 getCustomTextW6S22(  
 text: 'Search ${widget.selectedMethod}',  
 color: textWhiteColor,  
 ),  
 const Spacer()  
 ],  
 ),  
 getCustomTextFormField(  
 keyboardType: TextInputType.text,  
 obscureText: false,  
 hintText: 'Search (e.g. bank name)...'.tr,  
 controller: searchController,  
 onChanged: (value) {  
 searchedList.clear();  
 if (DataFile.bankMethodModel.isNotEmpty) {  
 for (var element in DataFile.bankMethodModel) {  
 if (element.nameBank!  
 .toLowerCase()  
 .contains(searchController.text)) {  
 searchedList.add(element);  
 }  
 }  
 }  
  
 if (searchController.text.isNotEmpty) {  
 isClosed.value = false;  
 } else {  
 isClosed.value = true;  
 }  
 },  
 prefixIcon: Icon(  
 Icons.search,  
 color: accentColor,  
 size: 30.px,  
 ),  
 suffixIcon: GestureDetector(  
 onTap: () {  
 searchController.clear();  
 searchedList.clear();  
 isClosed.value = true;  
 },  
 child: isClosed.value == true  
 ? Container()  
 : Icon(  
 Icons.cancel,  
 color: feePayCardColor,  
 size: 20.px,  
 )),  
 ),  
 ],  
 ),  
 ),  
 getVerSpace(3.h),  
 Expanded(  
 child: searchController.text.isNotEmpty  
 ? ListView.builder(  
 padding: EdgeInsets.symmetric(horizontal: 3.h),  
 shrinkWrap: true,  
 physics: const BouncingScrollPhysics(),  
 itemCount: searchedList.length,  
 itemBuilder: (context, index) {  
 return Obx(  
 () => GestureDetector(  
 onTap: () {  
 Get.to(() => AddBankScreen(  
 selectedBank: searchedList[index]  
 .nameBank  
 .toString(),  
 selectedMethod: widget.selectedMethod,  
 ));  
 Timer(const Duration(milliseconds: 150), () {  
 searchController.clear();  
 searchedList.clear();  
 isClosed.value = true;  
 });  
 },  
 child: Container(  
 margin: EdgeInsets.only(bottom: 1.5.h),  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius: BorderRadius.all(  
 Radius.circular(15.px)),  
 boxShadow: [  
 BoxShadow(  
 color: borderColor,  
 blurRadius: 5,  
 offset: const Offset(  
 0, 0), // changes position of shadow  
 ),  
 ],),  
 child: Column(  
 crossAxisAlignment:  
 CrossAxisAlignment.center,  
 mainAxisAlignment:  
 MainAxisAlignment.spaceBetween,  
 children: [  
 ListTile(  
 contentPadding: EdgeInsets.symmetric(  
 horizontal: 2.h),  
 leading: CircleAvatar(  
 radius: 2.5.h,  
 backgroundColor: borderColor),  
 title: getCustomTextW6S15(  
 text: searchedList[index]  
 .nameBank  
 .toString()  
 .tr,  
 color: textBlackColor),  
 ),  
 ],  
 ),  
 ),  
 ),  
 );  
 },  
 )  
 : ListView.builder(  
 shrinkWrap: true,  
 padding: EdgeInsets.symmetric(horizontal: 3.h),  
 physics: const BouncingScrollPhysics(),  
 itemCount: DataFile.bankMethodModel.length,  
 itemBuilder: (context, index) {  
 return GestureDetector(  
 onTap: () {  
 Get.to(() => AddBankScreen(  
 selectedBank: DataFile  
 .bankMethodModel[index].nameBank  
 .toString(),  
 selectedMethod: widget.selectedMethod,  
 ));  
 Timer(const Duration(milliseconds: 150), () {  
 searchController.clear();  
 searchedList.clear();  
 isClosed.value = true;  
 });  
 },  
 child: Container(  
 margin: EdgeInsets.only(bottom: 1.5.h),  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius: BorderRadius.all(  
 Radius.circular(15.px)),  
 boxShadow: [  
 BoxShadow(  
 color: borderColor,  
 blurRadius: 5,  
 offset: const Offset(  
 0, 0), // changes position of shadow  
 ),  
 ],),  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 mainAxisAlignment:  
 MainAxisAlignment.spaceBetween,  
 children: [  
 ListTile(  
 contentPadding:  
 EdgeInsets.symmetric(horizontal: 2.h),  
 leading: CircleAvatar(  
 radius: 2.5.h,  
 backgroundColor: borderColor),  
 title: getCustomTextW6S15(  
 text: DataFile  
 .bankMethodModel[index].nameBank  
 .toString()  
 .tr,  
 color: textBlackColor),  
 ),  
 ],  
 ),  
 ),  
 );  
 },  
 ),  
 )  
 ],  
 ),  
 ),  
 ),  
 ),  
 );  
 }  
}

# WalletScreens

## wallet\_screen

import 'package:algorthimi/view/WalletScreens/al\_ansari\_screen.dart';  
import 'package:algorthimi/view/WalletScreens/select\_bank.dart';  
import 'package:algorthimi/view/WalletScreens/withdraw\_crypto\_screen.dart';  
import 'package:algorthimi/view/WalletScreens/withdraw\_other\_method\_screen.dart';  
import 'package:flutter/material.dart';  
import 'package:flutter\_switch/flutter\_switch.dart';  
import 'package:get/get.dart';  
import 'package:responsive\_sizer/responsive\_sizer.dart';  
  
import '../../utils/color\_data.dart';  
import '../Widgets/widget\_utils.dart';  
import 'add\_card.dart';  
import 'congratulation\_screen.dart';  
  
class WalletScreen extends StatefulWidget {  
 const WalletScreen({Key? key}) : super(key: key);  
  
 @override  
 State<WalletScreen> createState() => \_WalletScreenState();  
}  
  
class \_WalletScreenState extends State<WalletScreen> {  
 var switchValue = true.obs;  
 var selectedAccount = 0.obs;  
 var selectedPaymentMethod = 0.obs;  
 var valueSlider = 100.0.obs;  
  
 @override  
 Widget build(BuildContext context) {  
 return Scaffold(  
 resizeToAvoidBottomInset: false,  
 backgroundColor: backgroundColor,  
 body: SizedBox(  
 height: 100.h,  
 width: 100.w,  
 child: Column(  
 children: [  
 SizedBox(  
 height: 45.h,  
 width: 100.h,  
 child: Stack(  
 children: [  
 Container(  
 height: 20.h,  
 width: 100.w,  
 alignment: Alignment.topCenter,  
 padding: EdgeInsets.only(left: 3.h, right: 3.h),  
 decoration: BoxDecoration(  
 color: accentColor,  
 borderRadius: BorderRadius.only(  
 bottomRight: Radius.circular(4.h))),  
 child: Row(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 mainAxisAlignment: MainAxisAlignment.start,  
 children: [  
 GestureDetector(  
 onTap: () {},  
 child: Container(),  
 ),  
 const Spacer(),  
 getCustomTextW6S22(  
 text: 'Wallet'.tr,  
 color: textWhiteColor,  
 ),  
 const Spacer()  
 ],  
 ),  
 ),  
 Positioned(  
 top: 15.h,  
 right: 0,  
 left: 0,  
 child: Obx(  
 () => Container(  
 width: 100.h,  
 height: 28.h,  
 padding: EdgeInsets.symmetric(  
 vertical: 2.h, horizontal: 3.h),  
 alignment: Alignment.topCenter,  
 margin: EdgeInsets.symmetric(horizontal: 3.h),  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 boxShadow: [  
 BoxShadow(  
 color: borderColor,  
 blurRadius: 1,  
 offset: const Offset(  
 0, 2), // changes position of shadow  
 ),  
 ],  
 borderRadius:  
 BorderRadius.all(Radius.circular(10.px))),  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.end,  
 mainAxisAlignment: MainAxisAlignment.spaceBetween,  
 children: [  
 Row(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 mainAxisAlignment: MainAxisAlignment.end,  
 children: [  
 getCustomTextW6S15(  
 text: 'Coins'.tr,  
 color: switchValue.value == false  
 ? pinkAppColor  
 : subTextColor),  
 getHorSpace(0.5.h),  
 FlutterSwitch(  
 height: 17.0,  
 width: 30.0,  
 padding: 2.0,  
 toggleSize: 12.0,  
 borderRadius: 10.0,  
 inactiveColor: pinkAppColor,  
 activeColor: accentColor,  
 value: switchValue.value,  
 onToggle: (value) {  
 switchValue.value = value;  
 selectedAccount.value = 0;  
 },  
 ),  
 getHorSpace(0.5.h),  
 getCustomTextW6S15(  
 text: 'Cash'.tr,  
 color: switchValue.value == true  
 ? accentColor  
 : subTextColor),  
 ],  
 ),  
 SizedBox(  
 width: 100.w,  
 child: Column(  
 mainAxisAlignment: MainAxisAlignment.center,  
 crossAxisAlignment: CrossAxisAlignment.center,  
 children: [  
 getCustomTextW6S15(  
 text: 'You currently have'.tr,  
 color: textBlackColor),  
 getCustomTextW6S30(  
 text: switchValue.value != true  
 ? '10 ${'Coins'.tr}'.tr  
 : 'AED 500.00'.tr,  
 color: switchValue.value != true  
 ? pinkAppColor  
 : accentColor),  
 switchValue.value == true  
 ? const SizedBox()  
 : getCustomTextW6S16(  
 text: '= AED 100'.tr,  
 color: subTextColor),  
 ]),  
 ),  
 switchValue.value == false  
 ? const SizedBox()  
 : SizedBox(  
 width: 100.w,  
 child: Column(  
 crossAxisAlignment:  
 CrossAxisAlignment.center,  
 mainAxisAlignment:  
 MainAxisAlignment.center,  
 children: [  
 SliderTheme(  
 data:  
 SliderTheme.of(context).copyWith(  
 activeTrackColor: accentColor,  
 inactiveTrackColor: borderColor,  
 trackShape:  
 const RoundedRectSliderTrackShape(),  
 trackHeight: 0.8.h,  
 thumbShape: RoundSliderThumbShape(  
 enabledThumbRadius: 1.0.h),  
 thumbColor: accentColor,  
 overlayColor: backgroundColor,  
 overlayShape:  
 RoundSliderOverlayShape(  
 overlayRadius: 0.h),  
 tickMarkShape:  
 const RoundSliderTickMarkShape(),  
 activeTickMarkColor: accentColor,  
 inactiveTickMarkColor: borderColor,  
 valueIndicatorShape:  
 const PaddleSliderValueIndicatorShape(),  
 valueIndicatorColor: accentColor,  
 valueIndicatorTextStyle: TextStyle(  
 color: textWhiteColor,  
 ),  
 ),  
 child: Slider(  
 value: valueSlider.value,  
 min: 100,  
 max: 1000,  
 divisions: 9,  
 label:  
 '${valueSlider.value.toInt()}',  
 onChanged: (value) {  
 valueSlider.value = value;  
 },  
 ),  
 ),  
 getVerSpace(0.5.h),  
 Padding(  
 padding: EdgeInsets.symmetric(  
 horizontal: 1.h),  
 child: Row(  
 mainAxisAlignment:  
 MainAxisAlignment.spaceBetween,  
 crossAxisAlignment:  
 CrossAxisAlignment.end,  
 children: [  
 getCustomTextW6S10(  
 text: 'AED 100',  
 color: subTextColor),  
 getCustomTextW6S10(  
 text: 'AED 1000',  
 color: subTextColor),  
 ],  
 ),  
 )  
 ],  
 ),  
 ),  
 ],  
 ),  
 ),  
 ),  
 ),  
 ],  
 ),  
 ),  
 Padding(  
 padding: EdgeInsets.symmetric(horizontal: 3.h),  
 child: Column(  
 children: [  
 getVerSpace(3.h),  
 GestureDetector(  
 onTap: () {  
 showModalBottomSheet<void>(  
 context: context,  
 elevation: 5,  
 builder: (BuildContext context) {  
 return Container(  
 height: 23.5.h,  
 width: 100.w,  
 padding: EdgeInsets.symmetric(  
 horizontal: 3.h, vertical: 2.h),  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius: BorderRadius.only(  
 topLeft: Radius.circular(15.px),  
 topRight: Radius.circular(15.px))),  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.start,  
 mainAxisAlignment:  
 MainAxisAlignment.spaceBetween,  
 children: [  
 Row(  
 crossAxisAlignment:  
 CrossAxisAlignment.center,  
 mainAxisAlignment:  
 MainAxisAlignment.spaceBetween,  
 children: [  
 getCustomTextW6S20(  
 text: 'Link Account'.tr,  
 color: accentColor),  
 InkWell(  
 onTap: () {  
 Get.back();  
 },  
 child: Icon(  
 Icons.close,  
 color: subTextColor,  
 size: 20,  
 ),  
 )  
 ],  
 ),  
 Column(  
 children: [  
 GestureDetector(  
 onTap: () {  
 Get.to(() => SelectBankScreen(  
 selectedMethod: 'Bank Account',  
 ));  
 },  
 child: fillColorButton(  
 text: 'Bank Account'.tr,  
 color: pinkAppColor),  
 ),  
 getVerSpace(1.4.h),  
 GestureDetector(  
 onTap: () {  
 Get.to(() => SelectBankScreen(  
 selectedMethod: 'E-Wallet',  
 ));  
 },  
 child: outlineButton(  
 text: 'E-Wallet'.tr,  
 color: backgroundColor,  
 borderColor: pinkAppColor)),  
 ],  
 )  
 ],  
 ),  
 );  
 },  
 );  
 },  
 child: fillColorButton(  
 text: 'Link Account'.tr, color: pinkAppColor)),  
 getVerSpace(1.5.h),  
 GestureDetector(  
 onTap: () {  
 showModalBottomSheet<void>(  
 context: context,  
 elevation: 5,  
 builder: (BuildContext context) {  
 return Container(  
 height: 50.h,  
 width: 100.w,  
 padding: EdgeInsets.only(  
 right: 3.h, left: 3.h, bottom: 4.h, top: 2.h),  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius: BorderRadius.only(  
 topLeft: Radius.circular(15.px),  
 topRight: Radius.circular(15.px))),  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.start,  
 mainAxisAlignment:  
 MainAxisAlignment.spaceBetween,  
 children: [  
 Column(  
 children: [  
 Row(  
 crossAxisAlignment:  
 CrossAxisAlignment.center,  
 mainAxisAlignment:  
 MainAxisAlignment.spaceBetween,  
 children: [  
 Expanded(  
 child: getCustomTextW6S20(  
 text: switchValue.value == true  
 ? 'Transfer to Bank/E-Wallet'  
 .tr  
 : 'Withdraw'.tr,  
 color: accentColor),  
 ),  
 InkWell(  
 onTap: () {  
 Get.back();  
 },  
 child: Icon(  
 Icons.close,  
 color: subTextColor,  
 size: 20,  
 ),  
 )  
 ],  
 ),  
 SizedBox(  
 height: 21.h,  
 child: ListView.builder(  
 shrinkWrap: true,  
 padding: EdgeInsets.symmetric(  
 vertical: 2.h),  
 itemCount: switchValue.value == true  
 ? withDrawTransferTitle.length  
 : withDrawTitle.length,  
 physics:  
 const BouncingScrollPhysics(),  
 itemBuilder: (context, index) {  
 return GestureDetector(  
 onTap: () {  
 selectedAccount.value = index;  
 },  
 child: Obx(  
 () => Container(  
 color: backgroundColor,  
 margin: EdgeInsets.only(  
 bottom: 1.1.h),  
 child: Column(  
 crossAxisAlignment:  
 CrossAxisAlignment  
 .start,  
 children: [  
 Row(  
 crossAxisAlignment:  
 CrossAxisAlignment  
 .center,  
 mainAxisAlignment:  
 MainAxisAlignment  
 .start,  
 children: [  
 Container(  
 width: 1.8.h,  
 height: 1.8.h,  
 decoration: BoxDecoration(  
 shape: BoxShape  
 .circle,  
 border: Border.all(  
 color: selectedAccount.value ==  
 index  
 ? accentColor  
 : borderColor,  
 width: selectedAccount.value ==  
 index  
 ? 4.px  
 : 1.5  
 .px)),  
 ),  
 getHorSpace(2.h),  
 Expanded(  
 child: getCustomTextW6S16(  
 text: switchValue  
 .value ==  
 true  
 ? withDrawTransferTitle[  
 index]  
 .toString()  
 .tr  
 : withDrawTitle[  
 index]  
 .toString()  
 .tr,  
 color: subTextColor,  
 textAlign: TextAlign.start  
 ),  
 ),  
 ],  
 ),  
 Row(  
 crossAxisAlignment:  
 CrossAxisAlignment  
 .center,  
 mainAxisAlignment:  
 MainAxisAlignment  
 .start,  
 children: [  
 SizedBox(  
 width: 1.8.h,  
 height: 1.8.h,  
 ),  
 getHorSpace(2.h),  
 Expanded(  
 child: getCustomTextW6S15(  
 text: switchValue  
 .value ==  
 true  
 ? withDrawTransferSubTitle[  
 index]  
 : withDrawSubTitle[  
 index]  
 .toString()  
 .tr,  
 color: subTextColor,  
 textAlign: TextAlign.start  
 ),  
 )  
 ],  
 ),  
 ],  
 ),  
 ),  
 ),  
 );  
 },  
 ),  
 ),  
 ],  
 ),  
 Column(  
 children: [  
 GestureDetector(  
 onTap: () {  
 switchValue.value == true  
 ? Get.to(() =>  
 CongratulationTransferBankEWalletScreen(  
 selectedMethod:  
 withDrawTransferTitle[  
 selectedAccount  
 .value]  
 .toString()  
 .tr,  
 ))  
 : selectedAccount.value == 1  
 ? Get.to(() =>  
 const WithdrawOtherMethodScreen())  
 : Get.to(() =>  
 const WithdrawCryptoScreen());  
 },  
 child: fillColorButton(  
 text: 'Transfer'.tr,  
 color: pinkAppColor),  
 ),  
 getVerSpace(1.4.h),  
 switchValue.value == false  
 ? Container()  
 : GestureDetector(  
 onTap: () {  
 Get.back();  
 showModalBottomSheet<void>(  
 context: context,  
 elevation: 5,  
 builder:  
 (BuildContext context) {  
 return Container(  
 height: 38.h,  
 width: 100.w,  
 padding: EdgeInsets.only(  
 right: 3.h,  
 left: 3.h,  
 bottom: 4.h,  
 top: 2.h),  
 decoration: BoxDecoration(  
 color:  
 backgroundColor,  
 borderRadius:  
 BorderRadius.only(  
 topLeft: Radius  
 .circular(15  
 .px),  
 topRight: Radius  
 .circular(  
 15.px))),  
 child: Column(  
 crossAxisAlignment:  
 CrossAxisAlignment  
 .start,  
 mainAxisAlignment:  
 MainAxisAlignment  
 .spaceBetween,  
 children: [  
 Column(  
 children: [  
 Row(  
 crossAxisAlignment:  
 CrossAxisAlignment  
 .center,  
 mainAxisAlignment:  
 MainAxisAlignment  
 .spaceBetween,  
 children: [  
 getCustomTextW6S20(  
 text: 'Add Payment Method'  
 .tr,  
 color:  
 accentColor),  
 InkWell(  
 onTap: () {  
 Get.back();  
 },  
 child: Icon(  
 Icons  
 .close,  
 color:  
 subTextColor,  
 size: 20,  
 ),  
 )  
 ],  
 ),  
 ListView.builder(  
 shrinkWrap:  
 true,  
 itemCount:  
 paymentMethodList  
 .length,  
 padding: EdgeInsets  
 .symmetric(  
 vertical:  
 2.h),  
 physics:  
 const NeverScrollableScrollPhysics(),  
 itemBuilder:  
 (context,  
 index) {  
 return GestureDetector(  
 onTap: () {  
 selectedPaymentMethod  
 .value =  
 index;  
 },  
 child: Obx(  
 () =>  
 Container(  
 color:  
 backgroundColor,  
 margin: EdgeInsets.only(  
 bottom:  
 2.h),  
 child:  
 Column(  
 crossAxisAlignment:  
 CrossAxisAlignment.start,  
 children: [  
 Row(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 mainAxisAlignment: MainAxisAlignment.start,  
 children: [  
 Container(  
 width: 1.8.h,  
 height: 1.8.h,  
 decoration: BoxDecoration(shape: BoxShape.circle, border: Border.all(color: selectedPaymentMethod.value == index ? accentColor : borderColor, width: selectedPaymentMethod.value == index ? 4.px : 1.5.px)),  
 ),  
 getHorSpace(2.h),  
 getCustomTextW6S16(  
 text: paymentMethodList[index].toString().tr,  
 color: subTextColor,  
 ),  
 ],  
 ),  
 ],  
 ),  
 ),  
 ),  
 );  
 },  
 ),  
 ],  
 ),  
 Column(  
 children: [  
 InkWell(  
 onTap: () {  
 Get.to(() => paymentMethodList[  
 selectedPaymentMethod.value] ==  
 "Credit/Debit Card"  
 ? AddCard(  
 selectedMethod:  
 paymentMethodList[selectedPaymentMethod.value],  
 )  
 : SelectBankScreen(  
 selectedMethod:  
 paymentMethodList[selectedPaymentMethod.value],  
 ));  
 },  
 child: fillColorButton(  
 text: 'Add'  
 .tr,  
 color:  
 pinkAppColor),  
 ),  
 ],  
 ),  
 ],  
 ),  
 );  
 },  
 );  
 },  
 child: outlineButton(  
 text: 'Add New'.tr,  
 borderColor: pinkAppColor,  
 color: backgroundColor)),  
 ],  
 ),  
 ],  
 ),  
 );  
 },  
 );  
 },  
 child: outlineButton(  
 text: 'Withdrawal Money'.tr,  
 color: backgroundColor,  
 borderColor: pinkAppColor)),  
 getVerSpace(1.5.h),  
 GestureDetector(  
 onTap: () {  
 Get.to(() => const AlAnsariExchangeScreen());  
 },  
 child: outlineButton(  
 text: 'Al Ansari Exchange'.tr,  
 color: backgroundColor,  
 borderColor: pinkAppColor),  
 )  
 ],  
 ),  
 ),  
 ],  
 ),  
 ),  
 );  
 }  
}  
  
List withDrawTitle = [  
 "Send via crypto network",  
 "Send via email/ Phone/ Pay ID",  
];  
List withDrawSubTitle = [  
 "send to a known crypto address\nvia crypto network",  
 "Recipient must be a Algorthmi user",  
];  
  
List withDrawTransferTitle = [  
 "First Abu Dhabi Bank",  
 "Dubai Islamic Bank",  
 "Etisalat Wallet",  
 "Magnati",  
];  
List withDrawTransferSubTitle = [  
 "xxx-xxxxx-xxxxxx",  
 "xxx-xxxxx-xxxxxx",  
 "xxx-xxxxx-xxxxxx",  
 "xxx-xxxxx-xxxxxx",  
];  
List paymentMethodList = [  
 // "Credit/Debit Card",  
 "Bank Account",  
 "E-Wallet",  
];

# WalletScreens

## withdraw\_crypto\_screen

import 'package:flutter/material.dart';  
import 'package:get/get.dart';  
import 'package:responsive\_sizer/responsive\_sizer.dart';  
  
import '../../utils/color\_data.dart';  
import '../Widgets/widget\_utils.dart';  
import 'congratulation\_screen.dart';  
  
class WithdrawCryptoScreen extends StatefulWidget {  
 const WithdrawCryptoScreen({Key? key}) : super(key: key);  
  
 @override  
 State<WithdrawCryptoScreen> createState() => \_WithdrawCryptoScreenState();  
}  
  
class \_WithdrawCryptoScreenState extends State<WithdrawCryptoScreen> {  
 TextEditingController coinController = TextEditingController();  
 TextEditingController addressController = TextEditingController();  
 TextEditingController networkController = TextEditingController();  
 TextEditingController amountController = TextEditingController();  
  
 back() {  
 Get.back();  
 Get.back();  
 }  
  
 @override  
 Widget build(BuildContext context) {  
 return WillPopScope(  
 onWillPop: () {  
 return back();  
 },  
 child: Scaffold(  
 resizeToAvoidBottomInset: false,  
 backgroundColor: backgroundColor,  
 body: SizedBox(  
 height: 100.h,  
 width: 100.h,  
 child: Stack(  
 children: [  
 Container(  
 height: 20.h,  
 width: 100.w,  
 alignment: Alignment.topCenter,  
 padding: EdgeInsets.only(top: 6.h, left: 3.h, right: 3.h),  
 decoration: BoxDecoration(  
 color: accentColor,  
 borderRadius:  
 BorderRadius.only(bottomRight: Radius.circular(4.h))),  
 child: Row(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 mainAxisAlignment: MainAxisAlignment.start,  
 children: [  
 GestureDetector(  
 onTap: () {  
 Get.back();  
 Get.back();  
 },  
 child: Icon(Icons.arrow\_back\_ios\_new\_outlined,  
 color: backgroundColor, size: 2.5.h),  
 ),  
 const Spacer(),  
 getCustomTextW6S22(  
 text: 'Withdraw Crypto',  
 color: textWhiteColor,  
 ),  
 const Spacer()  
 ],  
 ),  
 ),  
 Positioned(  
 top: 15.h,  
 right: 0,  
 left: 0,  
 child: Container(  
 width: 100.h,  
 height: 100.h - 20.h,  
 padding: EdgeInsets.only(top: 3.h, left: 2.h, right: 2.h),  
 alignment: Alignment.topCenter,  
 margin: EdgeInsets.symmetric(horizontal: 3.h),  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius: BorderRadius.all(Radius.circular(10.px))),  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 mainAxisAlignment: MainAxisAlignment.spaceBetween,  
 children: [  
 Column(  
 children: [  
 Row(  
 children: [  
 Expanded(  
 child: getCustomTextW6S15(  
 text: 'Select Coin', color: textBlackColor),  
 ),  
 getHorSpace(2.h),  
 Expanded(  
 flex: 3,  
 child: Container(  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius: BorderRadius.all(  
 Radius.circular(15.px)),  
 boxShadow: [  
 BoxShadow(  
 color: borderColor,  
 blurRadius: 10,  
 offset: const Offset(0,  
 0), // changes position of shadow  
 ),  
 ]),  
 child: getCustomTextFormField(  
 keyboardType: TextInputType.text,  
 obscureText: false,  
 hintText: 'Coin'.tr,  
 controller: coinController,  
 suffixIcon: Icon(  
 Icons.arrow\_drop\_down\_rounded,  
 color: subTextColor),  
 onChanged: (value) {},  
 ),  
 ),  
 ),  
 ],  
 ),  
 getVerSpace(1.5.h),  
 Row(  
 children: [  
 Expanded(  
 child: getCustomTextW6S15(  
 text: 'Withdraw to', color: textBlackColor),  
 ),  
 getHorSpace(2.h),  
 Expanded(  
 flex: 3,  
 child: Container(  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius: BorderRadius.all(  
 Radius.circular(15.px)),  
 boxShadow: [  
 BoxShadow(  
 color: borderColor,  
 blurRadius: 10,  
 offset: const Offset(0,  
 0), // changes position of shadow  
 ),  
 ]),  
 child: getCustomTextFormField(  
 keyboardType: TextInputType.text,  
 obscureText: false,  
 hintText: 'Address'.tr,  
 controller: addressController,  
 suffixIcon: Icon(  
 Icons.arrow\_drop\_down\_rounded,  
 color: subTextColor),  
 onChanged: (value) {},  
 ),  
 ),  
 ),  
 ],  
 ),  
 getVerSpace(1.5.h),  
 Row(  
 children: [  
 Expanded(  
 child: getCustomTextW6S15(  
 text: 'Network', color: textBlackColor),  
 ),  
 getHorSpace(2.h),  
 Expanded(  
 flex: 3,  
 child: Container(  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius: BorderRadius.all(  
 Radius.circular(15.px)),  
 boxShadow: [  
 BoxShadow(  
 color: borderColor,  
 blurRadius: 10,  
 offset: const Offset(0,  
 0), // changes position of shadow  
 ),  
 ]),  
 child: getCustomTextFormField(  
 keyboardType: TextInputType.text,  
 obscureText: false,  
 hintText: 'BTC Bitcoin (BTC)'.tr,  
 controller: networkController,  
 suffixIcon: Icon(  
 Icons.arrow\_drop\_down\_rounded,  
 color: subTextColor),  
 onChanged: (value) {},  
 ),  
 ),  
 ),  
 ],  
 ),  
 getVerSpace(0.5.h),  
 Row(  
 children: [  
 Expanded(  
 child: getCustomTextW6S15(  
 text: '', color: textBlackColor),  
 ),  
 getHorSpace(2.h),  
 Expanded(  
 flex: 3,  
 child: getCustomTextW6S12(  
 text:  
 'Wallet address automatically matched to corresponding network',  
 color: greenColor),  
 ),  
 ],  
 ),  
 getVerSpace(1.5.h),  
 Row(  
 children: [  
 Expanded(  
 child: getCustomTextW6S15(  
 text: '', color: textBlackColor),  
 ),  
 getHorSpace(2.h),  
 Expanded(  
 flex: 3,  
 child: getCustomTextW6S12(  
 text: 'BTC available',  
 color: subTextColor,  
 textAlign: TextAlign.right),  
 ),  
 getHorSpace(1.h),  
 ],  
 ),  
 getVerSpace(0.5.h),  
 Row(  
 children: [  
 Expanded(  
 child: getCustomTextW6S15(  
 text: 'Withdraw Amount',  
 color: textBlackColor),  
 ),  
 getHorSpace(2.h),  
 Expanded(  
 flex: 3,  
 child: Container(  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius: BorderRadius.all(  
 Radius.circular(15.px)),  
 boxShadow: [  
 BoxShadow(  
 color: borderColor,  
 blurRadius: 10,  
 offset: const Offset(0,  
 0), // changes position of shadow  
 ),  
 ]),  
 child: getCustomTextFormField(  
 keyboardType: TextInputType.text,  
 obscureText: false,  
 hintText: 'Enter Amount'.tr,  
 controller: amountController,  
 suffixIconConstraintsWidth: 10.h,  
 suffixIcon: getCustomTextW6S12(  
 text: 'MAX BTC',  
 color: textBlackColor),  
 onChanged: (value) {},  
 ),  
 ),  
 ),  
 ],  
 ),  
 getVerSpace(0.5.h),  
 Row(  
 children: [  
 Expanded(  
 child: getCustomTextW6S15(  
 text: '', color: textBlackColor),  
 ),  
 getHorSpace(2.h),  
 Expanded(  
 flex: 3,  
 child: getCustomTextW6S12(  
 text:  
 '100 BTC /100 BTC 24h remaining limit ',  
 color: greenColor),  
 ),  
 ],  
 ),  
 ],  
 ),  
 Column(  
 mainAxisAlignment: MainAxisAlignment.center,  
 crossAxisAlignment: CrossAxisAlignment.center,  
 children: [  
 getCustomTextW6S15(  
 text: 'Receive amount', color: textBlackColor),  
 getCustomTextW6S36(  
 text: '0.99950000 BTC', color: pinkAppColor),  
 getCustomTextW6S12(  
 text: '0.0005 BTC network fee included',  
 color: subTextColor)  
 ],  
 ),  
 GestureDetector(  
 onTap: () {  
 Get.to(() => CongratulationTransferBankEWalletScreen(  
 selectedMethod: 'withdrawCrypto',  
 ));  
 },  
 child: fillColorButton(  
 color: pinkAppColor, text: 'Withdraw'),  
 ),  
 ],  
 ),  
 ),  
 ),  
 ],  
 ),  
 ),  
 ),  
 );  
 }  
}

# WalletScreens

## withdraw\_other\_method\_screen

import 'package:algorthimi/view/WalletScreens/withdraw\_payment\_verification\_screen.dart';  
import 'package:flutter/material.dart';  
import 'package:get/get.dart';  
import 'package:responsive\_sizer/responsive\_sizer.dart';  
  
import '../../utils/color\_data.dart';  
import '../Widgets/widget\_utils.dart';  
  
class WithdrawOtherMethodScreen extends StatefulWidget {  
 const WithdrawOtherMethodScreen({Key? key}) : super(key: key);  
  
 @override  
 State<WithdrawOtherMethodScreen> createState() =>  
 \_WithdrawOtherMethodScreenState();  
}  
  
class \_WithdrawOtherMethodScreenState extends State<WithdrawOtherMethodScreen> {  
 TextEditingController coinController = TextEditingController();  
 TextEditingController addressController = TextEditingController();  
 TextEditingController noteController = TextEditingController();  
 TextEditingController amountController = TextEditingController();  
  
 back() {  
 Get.back();  
 Get.back();  
 }  
  
 @override  
 Widget build(BuildContext context) {  
 return WillPopScope(  
 onWillPop: () {  
 return back();  
 },  
 child: Scaffold(  
 resizeToAvoidBottomInset: false,  
 backgroundColor: backgroundColor,  
 body: SingleChildScrollView(  
 child: SizedBox(  
 height: 100.h,  
 width: 100.h,  
 child: Stack(  
 children: [  
 Container(  
 height: 16.h,  
 width: 100.w,  
 alignment: Alignment.topCenter,  
 padding: EdgeInsets.only(top: 6.h, left: 3.h, right: 3.h),  
 decoration: BoxDecoration(  
 color: accentColor,  
 borderRadius:  
 BorderRadius.only(bottomRight: Radius.circular(4.h))),  
 child: Row(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 mainAxisAlignment: MainAxisAlignment.start,  
 children: [  
 GestureDetector(  
 onTap: () {  
 Get.back();  
 Get.back();  
 },  
 child: Icon(Icons.arrow\_back\_ios\_new\_outlined,  
 color: backgroundColor, size: 2.5.h),  
 ),  
 const Spacer(),  
 getCustomTextW6S22(  
 text: 'Withdraw Crypto',  
 color: textWhiteColor,  
 ),  
 const Spacer()  
 ],  
 ),  
 ),  
 Positioned(  
 top: 12.5.h,  
 right: 0,  
 left: 0,  
 child: Container(  
 width: 100.h,  
 height: 100.h - 16.h,  
 padding: EdgeInsets.only(top: 1.h, left: 2.h, right: 2.h),  
 alignment: Alignment.topCenter,  
 margin: EdgeInsets.symmetric(horizontal: 3.h),  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius: BorderRadius.all(Radius.circular(10.px))),  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 mainAxisAlignment: MainAxisAlignment.spaceBetween,  
 children: [  
 Column(  
 children: [  
 SizedBox(  
 width: 100.w,  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.start,  
 children: [  
 getCustomTextW6S22(  
 text: 'Send BTC', color: textBlackColor),  
 getCustomTextW6S15(  
 text: 'Send BTC to Binance account',  
 color: subTextColor)  
 ],  
 ),  
 ),  
 getVerSpace(2.h),  
 Row(  
 children: [  
 Expanded(  
 child: getCustomTextW6S15(  
 text: 'Send Mode', color: textBlackColor),  
 ),  
 getHorSpace(2.h),  
 Expanded(  
 flex: 3,  
 child: Container(  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius: BorderRadius.all(  
 Radius.circular(15.px)),  
 boxShadow: [  
 BoxShadow(  
 color: borderColor,  
 blurRadius: 10,  
 offset: const Offset(0,  
 0), // changes position of shadow  
 ),  
 ]),  
 child: getCustomTextFormField(  
 keyboardType: TextInputType.text,  
 obscureText: false,  
 hintText: 'Email'.tr,  
 controller: coinController,  
 suffixIcon: Icon(  
 Icons.arrow\_drop\_down\_rounded,  
 color: subTextColor),  
 onChanged: (value) {},  
 ),  
 ),  
 ),  
 ],  
 ),  
 getVerSpace(1.5.h),  
 Row(  
 children: [  
 Expanded(  
 child: getCustomTextW6S15(  
 text: 'Email', color: textBlackColor),  
 ),  
 getHorSpace(2.h),  
 Expanded(  
 flex: 3,  
 child: Container(  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius: BorderRadius.all(  
 Radius.circular(15.px)),  
 boxShadow: [  
 BoxShadow(  
 color: borderColor,  
 blurRadius: 10,  
 offset: const Offset(0,  
 0), // changes position of shadow  
 ),  
 ]),  
 child: getCustomTextFormField(  
 keyboardType: TextInputType.text,  
 obscureText: false,  
 hintText: '12354@gmail.com'.tr,  
 controller: addressController,  
 onChanged: (value) {},  
 ),  
 ),  
 ),  
 ],  
 ),  
 getVerSpace(1.5.h),  
 Row(  
 children: [  
 Expanded(  
 child: getCustomTextW6S15(  
 text: 'Amount', color: textBlackColor),  
 ),  
 getHorSpace(2.h),  
 Expanded(  
 flex: 3,  
 child: Container(  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius: BorderRadius.all(  
 Radius.circular(15.px)),  
 boxShadow: [  
 BoxShadow(  
 color: borderColor,  
 blurRadius: 10,  
 offset: const Offset(0,  
 0), // changes position of shadow  
 ),  
 ]),  
 child: getCustomTextFormField(  
 keyboardType: TextInputType.text,  
 obscureText: false,  
 hintText: 'Enter Amount'.tr,  
 controller: amountController,  
 suffixIconConstraintsWidth: 10.h,  
 suffixIcon: getCustomTextW6S12(  
 text: 'MAX BTC',  
 color: textBlackColor),  
 onChanged: (value) {},  
 ),  
 ),  
 ),  
 ],  
 ),  
 getVerSpace(1.5.h),  
 Container(  
 width: 100.w,  
 decoration: BoxDecoration(  
 borderRadius:  
 BorderRadius.all(Radius.circular(10.px)),  
 border: Border.all(  
 color: accentColor, width: 1.px)),  
 child: Column(  
 children: [  
 Container(  
 height: 5.h,  
 width: 100.w,  
 decoration: BoxDecoration(  
 color: accentColor,  
 border: Border.all(  
 color: accentColor, width: 1.px),  
 borderRadius: BorderRadius.all(  
 Radius.circular(8.px))),  
 child: Row(  
 crossAxisAlignment:  
 CrossAxisAlignment.center,  
 mainAxisAlignment:  
 MainAxisAlignment.spaceAround,  
 children: [  
 getCustomTextW6S15(  
 text: 'Available',  
 color: textWhiteColor),  
 getCustomTextW6S15(  
 text: '9.32 BTC',  
 color: textWhiteColor),  
 ],  
 )),  
 Container(  
 margin: EdgeInsets.symmetric(  
 horizontal: 2.h, vertical: 2.h),  
 child: Column(  
 children: [  
 Padding(  
 padding: EdgeInsets.only(bottom: 1.h),  
 child: Row(  
 crossAxisAlignment:  
 CrossAxisAlignment.center,  
 mainAxisAlignment:  
 MainAxisAlignment.spaceBetween,  
 children: [  
 Row(  
 children: [  
 Icon(Icons.check\_circle,  
 color: greenColor,  
 size: 2.5.h),  
 getHorSpace(1.h),  
 getCustomTextW6S12(  
 text: 'Spot Wallet',  
 color: subTextColor),  
 ],  
 ),  
 getCustomTextW6S12(  
 text: '1 BTC',  
 color: subTextColor)  
 ],  
 ),  
 ),  
 Row(  
 crossAxisAlignment:  
 CrossAxisAlignment.center,  
 mainAxisAlignment:  
 MainAxisAlignment.spaceBetween,  
 children: [  
 Row(  
 children: [  
 Icon(Icons.check\_circle,  
 color: greenColor,  
 size: 2.5.h),  
 getHorSpace(1.h),  
 getCustomTextW6S12(  
 text: 'Funding Wallet',  
 color: subTextColor),  
 ],  
 ),  
 getCustomTextW6S12(  
 text: '0 BTC',  
 color: subTextColor)  
 ],  
 ),  
 ],  
 ),  
 )  
 ],  
 ),  
 ),  
 getVerSpace(1.5.h),  
 SizedBox(  
 width: 100.w,  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.start,  
 children: [  
 getCustomTextW6S15(  
 text: 'Note (Optinal)',  
 color: textBlackColor),  
 getVerSpace(1.h),  
 Container(  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius: BorderRadius.all(  
 Radius.circular(15.px)),  
 boxShadow: [  
 BoxShadow(  
 color: borderColor,  
 blurRadius: 10,  
 offset: const Offset(0,  
 0), // changes position of shadow  
 ),  
 ]),  
 child: getCustomTextFormField(  
 keyboardType: TextInputType.text,  
 obscureText: false,  
 hintText:  
 'Add a note for the recipient'.tr,  
 controller: noteController,  
 suffixIcon: getCustomTextW6S12(  
 text: '0/50', color: textBlackColor),  
 onChanged: (value) {},  
 ),  
 ),  
 ],  
 ),  
 )  
 ],  
 ),  
 Column(  
 mainAxisAlignment: MainAxisAlignment.center,  
 crossAxisAlignment: CrossAxisAlignment.center,  
 children: [  
 getCustomTextW6S15(  
 text: 'Total Coins', color: textBlackColor),  
 getCustomTextW6S36(  
 text: '1 BTC', color: pinkAppColor),  
 getCustomTextW6S12(  
 text: 'No fees', color: subTextColor)  
 ],  
 ),  
 GestureDetector(  
 onTap: () {  
 showModalBottomSheet<void>(  
 context: context,  
 elevation: 5,  
 builder: (BuildContext context) {  
 return Container(  
 width: 100.w,  
 padding: EdgeInsets.symmetric(  
 vertical: 2.h, horizontal: 3.h),  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius: BorderRadius.only(  
 topLeft: Radius.circular(15.px),  
 topRight: Radius.circular(15.px))),  
 child: Column(  
 crossAxisAlignment:  
 CrossAxisAlignment.center,  
 mainAxisAlignment:  
 MainAxisAlignment.spaceBetween,  
 children: [  
 Column(  
 children: [  
 Row(  
 crossAxisAlignment:  
 CrossAxisAlignment.start,  
 mainAxisAlignment:  
 MainAxisAlignment.spaceBetween,  
 children: [  
 getCustomTextW6S17(  
 text: 'Confirm Order'.tr,  
 color: greenColor),  
 InkWell(  
 onTap: () {  
 Get.back();  
 },  
 child: Icon(  
 Icons.close,  
 color: subTextColor,  
 size: 20,  
 ),  
 )  
 ],  
 ),  
 getVerSpace(2.h),  
 Column(  
 mainAxisAlignment:  
 MainAxisAlignment.center,  
 crossAxisAlignment:  
 CrossAxisAlignment.center,  
 children: [  
 getCustomTextW6S15(  
 text: 'Total Coins',  
 color: textBlackColor),  
 getCustomTextW6S36(  
 text: '1 BTC',  
 color: pinkAppColor),  
 getCustomTextW6S12(  
 text: '= \$ 110,12.25',  
 color: subTextColor)  
 ],  
 ),  
 getVerSpace(2.h),  
 Container(  
 width: 100.w,  
 decoration: BoxDecoration(  
 color: subTextColor  
 .withOpacity(0.2),  
 borderRadius: BorderRadius.all(  
 Radius.circular(10.px))),  
 padding: EdgeInsets.symmetric(  
 horizontal: 2.h, vertical: 1.h),  
 child: Column(  
 children: [  
 Row(  
 crossAxisAlignment:  
 CrossAxisAlignment.center,  
 mainAxisAlignment:  
 MainAxisAlignment  
 .spaceBetween,  
 children: [  
 getCustomTextW6S12(  
 text: 'Send Mode',  
 color: subTextColor),  
 getCustomTextW6S12(  
 text: 'Email',  
 color: textBlackColor),  
 ],  
 ),  
 getVerSpace(0.4.h),  
 Row(  
 crossAxisAlignment:  
 CrossAxisAlignment.center,  
 mainAxisAlignment:  
 MainAxisAlignment  
 .spaceBetween,  
 children: [  
 getCustomTextW6S12(  
 text: 'To',  
 color: subTextColor),  
 getCustomTextW6S12(  
 text:  
 '12345@gmail.com ',  
 color: textBlackColor),  
 ],  
 ),  
 getVerSpace(0.4.h),  
 Row(  
 crossAxisAlignment:  
 CrossAxisAlignment.center,  
 mainAxisAlignment:  
 MainAxisAlignment  
 .spaceBetween,  
 children: [  
 getCustomTextW6S12(  
 text: 'Amount',  
 color: subTextColor),  
 getCustomTextW6S12(  
 text: '1 BCD',  
 color: textBlackColor),  
 ],  
 ),  
 getVerSpace(0.4.h),  
 Row(  
 crossAxisAlignment:  
 CrossAxisAlignment.center,  
 mainAxisAlignment:  
 MainAxisAlignment  
 .spaceBetween,  
 children: [  
 getCustomTextW6S12(  
 text: 'Send from',  
 color: subTextColor),  
 getCustomTextW6S12(  
 text: 'Spot Wallet',  
 color: textBlackColor),  
 ],  
 ),  
 getVerSpace(0.4.h),  
 Row(  
 crossAxisAlignment:  
 CrossAxisAlignment.center,  
 mainAxisAlignment:  
 MainAxisAlignment  
 .spaceBetween,  
 children: [  
 getCustomTextW6S12(  
 text: 'Note',  
 color: subTextColor),  
 getCustomTextW6S12(  
 text: 'Send it!',  
 color: textBlackColor),  
 ],  
 ),  
 ],  
 ),  
 ),  
 getVerSpace(2.h),  
 Container(  
 width: 100.w,  
 decoration: BoxDecoration(  
 color: subTextColor  
 .withOpacity(0.2),  
 borderRadius: BorderRadius.all(  
 Radius.circular(10.px))),  
 padding: EdgeInsets.symmetric(  
 horizontal: 2.h, vertical: 1.h),  
 child: getCustomTextW6S12(  
 text:  
 'Transactions are limited to crypto transfers between Binance accounts. Enjoy instant arrival with no fees. Refunds are not supported.',  
 color: textBlackColor),  
 )  
 ],  
 ),  
 GestureDetector(  
 onTap: () {  
 Get.to(() =>  
 const WithdrawPaymentVerificationScreen());  
 },  
 child: fillColorButton(  
 color: pinkAppColor, text: 'Next'),  
 ),  
 ],  
 ),  
 );  
 },  
 );  
 },  
 child: fillColorButton(  
 color: pinkAppColor, text: 'Send'),  
 ),  
 ],  
 ),  
 ),  
 ),  
 ],  
 ),  
 ),  
 ),  
 ),  
 );  
 }  
}

# WalletScreens

## withdraw\_payment\_verification\_screen

import 'package:flutter/material.dart';  
import 'package:get/get.dart';  
import 'package:responsive\_sizer/responsive\_sizer.dart';  
  
import '../../utils/color\_data.dart';  
import '../BottomNavBar/bottom\_nav\_bar\_freelancer\_screen.dart';  
import '../Widgets/widget\_utils.dart';  
  
class WithdrawPaymentVerificationScreen extends StatefulWidget {  
 const WithdrawPaymentVerificationScreen({Key? key}) : super(key: key);  
  
 @override  
 State<WithdrawPaymentVerificationScreen> createState() =>  
 \_WithdrawPaymentVerificationScreenState();  
}  
  
class \_WithdrawPaymentVerificationScreenState  
 extends State<WithdrawPaymentVerificationScreen> {  
 TextEditingController phoneVerificationController = TextEditingController();  
 TextEditingController emailVerificationController = TextEditingController();  
 TextEditingController googleVerificationController = TextEditingController();  
  
 back() {  
 Get.back();  
 Get.back();  
 }  
  
 @override  
 Widget build(BuildContext context) {  
 return WillPopScope(  
 onWillPop: () {  
 return back();  
 },  
 child: Scaffold(  
 resizeToAvoidBottomInset: false,  
 backgroundColor: backgroundColor,  
 body: SizedBox(  
 height: 100.h,  
 width: 100.h,  
 child: Stack(  
 children: [  
 Container(  
 height: 20.h,  
 width: 100.w,  
 alignment: Alignment.topCenter,  
 padding: EdgeInsets.only(top: 6.h, left: 3.h, right: 3.h),  
 decoration: BoxDecoration(  
 color: accentColor,  
 borderRadius:  
 BorderRadius.only(bottomRight: Radius.circular(4.h))),  
 child: Row(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 mainAxisAlignment: MainAxisAlignment.start,  
 children: [  
 GestureDetector(  
 onTap: () {  
 Get.back();  
 Get.back();  
 },  
 child: Icon(Icons.arrow\_back\_ios\_new\_outlined,  
 color: backgroundColor, size: 2.5.h),  
 ),  
 const Spacer(),  
 getCustomTextW6S22(  
 text: 'Verification',  
 color: textWhiteColor,  
 ),  
 const Spacer()  
 ],  
 ),  
 ),  
 Positioned(  
 top: 15.h,  
 right: 0,  
 left: 0,  
 child: Container(  
 width: 100.h,  
 height: 100.h - 20.h,  
 padding: EdgeInsets.only(top: 3.h, left: 2.h, right: 2.h),  
 alignment: Alignment.topCenter,  
 margin: EdgeInsets.symmetric(horizontal: 3.h),  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius: BorderRadius.all(Radius.circular(10.px))),  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.start,  
 mainAxisAlignment: MainAxisAlignment.spaceBetween,  
 children: [  
 Column(  
 crossAxisAlignment: CrossAxisAlignment.start,  
 children: [  
 getCustomTextW6S12(  
 text:  
 'Mobile code has been send to +971 55 \*\*\*\*\*\*\*\*',  
 color: subTextColor),  
 getVerSpace(1.h),  
 Container(  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius:  
 BorderRadius.all(Radius.circular(15.px)),  
 boxShadow: [  
 BoxShadow(  
 color: borderColor,  
 blurRadius: 10,  
 offset: const Offset(  
 0, 0), // changes position of shadow  
 ),  
 ]),  
 child: getCustomTextFormField(  
 keyboardType: TextInputType.text,  
 obscureText: false,  
 hintText: 'Mobile verification code'.tr,  
 suffixIcon: getCustomTextW6S15(  
 text: 'Send Code', color: pinkAppColor),  
 suffixIconConstraintsWidth: 12.h,  
 controller: phoneVerificationController,  
 onChanged: (value) {},  
 ),  
 ),  
 getVerSpace(2.h),  
 getCustomTextW6S12(  
 text:  
 'Email code will be sent to 123\*\*@\*\*\*\*\*.com',  
 color: subTextColor),  
 getVerSpace(1.h),  
 Container(  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius:  
 BorderRadius.all(Radius.circular(15.px)),  
 boxShadow: [  
 BoxShadow(  
 color: borderColor,  
 blurRadius: 10,  
 offset: const Offset(  
 0, 0), // changes position of shadow  
 ),  
 ]),  
 child: getCustomTextFormField(  
 keyboardType: TextInputType.text,  
 obscureText: false,  
 hintText: 'Email verification code '.tr,  
 suffixIcon: getCustomTextW6S15(  
 text: 'Send Code', color: pinkAppColor),  
 suffixIconConstraintsWidth: 12.h,  
 controller: emailVerificationController,  
 onChanged: (value) {},  
 ),  
 ),  
 getVerSpace(2.h),  
 getCustomTextW6S12(  
 text: 'Google verification code',  
 color: subTextColor),  
 getVerSpace(1.h),  
 Container(  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius:  
 BorderRadius.all(Radius.circular(15.px)),  
 boxShadow: [  
 BoxShadow(  
 color: borderColor,  
 blurRadius: 10,  
 offset: const Offset(  
 0, 0), // changes position of shadow  
 ),  
 ]),  
 child: getCustomTextFormField(  
 keyboardType: TextInputType.text,  
 obscureText: false,  
 hintText: 'Google verification code'.tr,  
 suffixIcon: getCustomTextW6S15(  
 text: 'Send Code', color: pinkAppColor),  
 suffixIconConstraintsWidth: 12.h,  
 controller: googleVerificationController,  
 onChanged: (value) {},  
 ),  
 ),  
 ],  
 ),  
 Column(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 mainAxisAlignment: MainAxisAlignment.center,  
 children: [  
 GestureDetector(  
 onTap: () {  
 Get.to(() => const BottomNavBarFreelancerScreen());  
 },  
 child: fillColorButton(  
 color: pinkAppColor, text: 'Submit'),  
 ),  
 getVerSpace(1.5.h),  
 getCustomTextW6S12(  
 text: 'Security verification unavailable?',  
 color: subTextColor)  
 ],  
 ),  
 ],  
 ),  
 ),  
 ),  
 ],  
 ),  
 ),  
 ),  
 );  
 }  
}

# Widgets

## firebase\_otp

import 'package:firebase\_core/firebase\_core.dart';  
import 'package:flutter/material.dart';  
import 'package:firebase\_auth/firebase\_auth.dart';  
  
  
  
class OTPScreen extends StatefulWidget {  
 @override  
 \_OTPScreenState createState() => \_OTPScreenState();  
}  
  
class \_OTPScreenState extends State<OTPScreen> {  
 final TextEditingController \_phoneNumberController = TextEditingController();  
 final TextEditingController \_otpController = TextEditingController();  
 FirebaseAuth \_auth = FirebaseAuth.instance;  
 String \_verificationId = '';  
  
 Future<void> verifyPhoneNumber(BuildContext context) async {  
 PhoneVerificationCompleted verificationCompleted =  
 (PhoneAuthCredential phoneAuthCredential) async {  
 await \_auth.signInWithCredential(phoneAuthCredential);  
 ScaffoldMessenger.of(context).showSnackBar(  
 SnackBar(  
 content: Text("Phone number automatically verified."),  
 ),  
 );  
 };  
  
 PhoneVerificationFailed verificationFailed =  
 (FirebaseAuthException authException) {  
 ScaffoldMessenger.of(context).showSnackBar(  
 SnackBar(  
 content: Text(  
 'Phone number verification failed. Code: ${authException.code}. Message: ${authException.message}'),  
 ),  
 );  
 };  
  
 PhoneCodeSent codeSent =  
 (String verificationId, [int? forceResendingToken]) async {  
 ScaffoldMessenger.of(context).showSnackBar(  
 SnackBar(  
 content: Text('Please check your phone for the verification code.'),  
 ),  
 );  
 \_verificationId = verificationId;  
 };  
  
 PhoneCodeAutoRetrievalTimeout codeAutoRetrievalTimeout =  
 (String verificationId) {  
 \_verificationId = verificationId;  
 };  
  
 try {  
 await \_auth.verifyPhoneNumber(  
 phoneNumber: \_phoneNumberController.text,  
 timeout: const Duration(seconds: 60),  
 verificationCompleted: verificationCompleted,  
 verificationFailed: verificationFailed,  
 codeSent: codeSent,  
 codeAutoRetrievalTimeout: codeAutoRetrievalTimeout,  
 );  
 } catch (e) {  
 ScaffoldMessenger.of(context).showSnackBar(  
 SnackBar(  
 content: Text("Failed to Verify Phone Number: $e"),  
 ),  
 );  
 }  
 }  
  
 void signInWithPhoneNumber(BuildContext context) async {  
 try {  
 final AuthCredential credential = PhoneAuthProvider.credential(  
 verificationId: \_verificationId,  
 smsCode: \_otpController.text,  
 );  
 await \_auth.signInWithCredential(credential);  
 ScaffoldMessenger.of(context).showSnackBar(  
 SnackBar(  
 content: Text("Phone number successfully verified."),  
 ),  
 );  
 } catch (e) {  
 ScaffoldMessenger.of(context).showSnackBar(  
 SnackBar(  
 content: Text("Failed to verify phone number: $e"),  
 ),  
 );  
 }  
 }  
  
 @override  
 Widget build(BuildContext context) {  
 return Scaffold(  
 appBar: AppBar(  
 title: Text("OTP Authentication"),  
 ),  
  
  
 body: Padding(  
 padding: const EdgeInsets.all(20.0),  
 child: Column(  
 mainAxisAlignment: MainAxisAlignment.center,  
 children: <Widget>[  
 TextField(  
 controller: \_phoneNumberController,  
 decoration: InputDecoration(labelText: 'Phone number'),  
 ),  
 SizedBox(height: 20),  
 ElevatedButton(  
 onPressed: () => verifyPhoneNumber(context),  
 child: Text("Send OTP"),  
 ),  
 SizedBox(height: 20),  
  
  
  
  
  
 TextField(  
 controller: \_otpController,  
 decoration: InputDecoration(labelText: 'Enter OTP'),  
 ),  
 SizedBox(height: 20),  
 ElevatedButton(  
 onPressed: () => signInWithPhoneNumber(context),  
 child: Text("Verify OTP"),  
 ),  
 ],  
 ),  
 ),  
 );  
 }  
}

# Widgets

## shop\_area\_find

import 'dart:async';  
  
import 'package:flutter/material.dart';  
import 'package:pedometer/pedometer.dart';  
  
  
class StepCounterScreen extends StatefulWidget {  
 @override  
 \_StepCounterScreenState createState() => \_StepCounterScreenState();  
}  
  
class \_StepCounterScreenState extends State<StepCounterScreen> {  
 late Stream<StepCount> \_stepCountStream;  
 int \_totalSteps = 0;  
 bool \_isCountingSteps = false;  
 Stopwatch \_stopwatch = Stopwatch();  
 late Timer \_timer;  
  
 // Conversion factor: 1 meter^2 = 10.7639 square feet  
 final double squareFeetPerSquareMeter = 10.7639;  
 // Conversion factor: 1 square meter = 0.005 marlas  
 final double marlasPerSquareMeter = 0.005;  
  
 // Assuming an average stride length of 0.75 meters  
 final double averageStrideLengthMeters = 0.75;  
  
 @override  
 void initState() {  
 super.initState();  
 \_initPedometer();  
 }  
  
 void \_initPedometer() {  
 \_stepCountStream = Pedometer.stepCountStream;  
 \_stepCountStream.listen((StepCount event) {  
 if (\_isCountingSteps) {  
 setState(() {  
 \_totalSteps = event.steps;  
 });  
 }  
 });  
 }  
  
 void \_startCountingSteps() {  
 setState(() {  
 \_isCountingSteps = true;  
 \_stopwatch.start();  
 \_startTimer();  
 });  
 }  
  
 void \_stopCountingSteps() {  
 setState(() {  
 \_isCountingSteps = false;  
 \_stopwatch.stop();  
 \_stopTimer();  
 });  
 }  
  
 void \_resetTime() {  
 setState(() {  
 \_stopwatch.reset();  
 \_totalSteps = 0;  
 });  
 }  
  
 void \_startTimer() {  
 \_timer = Timer.periodic(Duration(seconds: 1), (timer) {  
 setState(() {});  
 });  
 }  
  
 void \_stopTimer() {  
 \_timer.cancel();  
 }  
  
 String \_formatElapsedTime(Duration duration) {  
 final minutes = duration.inMinutes;  
 final seconds = duration.inSeconds % 60;  
 return '$minutes:${seconds.toString().padLeft(2, '0')}';  
 }  
  
 double \_calculateDistanceInSquareMeters(int steps) {  
 return steps \* averageStrideLengthMeters;  
 }  
  
 double \_convertSquareMetersToSquareFeet(double squareMeters) {  
 return squareMeters \* squareFeetPerSquareMeter;  
 }  
  
 double \_convertSquareMetersToMarlas(double squareMeters) {  
 return squareMeters \* marlasPerSquareMeter;  
 }  
  
 @override  
 Widget build(BuildContext context) {  
 double distanceInSquareMeters = \_calculateDistanceInSquareMeters(\_totalSteps);  
 double distanceInSquareFeet = \_convertSquareMetersToSquareFeet(distanceInSquareMeters);  
 double distanceInMarlas = \_convertSquareMetersToMarlas(distanceInSquareMeters);  
  
 return Scaffold(  
 appBar: AppBar(title: Text('Step Counter')),  
 body: Center(  
 child: Column(  
 mainAxisAlignment: MainAxisAlignment.center,  
 children: [  
 Text(  
 'Total Steps:',  
 style: TextStyle(fontSize: 20),  
 ),  
 Text(  
 '$\_totalSteps',  
 style: TextStyle(fontSize: 40, fontWeight: FontWeight.bold),  
 ),  
 SizedBox(height: 20),  
 Text(  
 'Total Distance (Square Feet):',  
 style: TextStyle(fontSize: 16),  
 ),  
 Text(  
 '${distanceInSquareFeet.toStringAsFixed(2)}',  
 style: TextStyle(fontSize: 24, fontWeight: FontWeight.bold),  
 ),  
 SizedBox(height: 10),  
 Text(  
 'Total Distance (Marlas):',  
 style: TextStyle(fontSize: 16),  
 ),  
 Text(  
 '${distanceInMarlas.toStringAsFixed(2)}',  
 style: TextStyle(fontSize: 24, fontWeight: FontWeight.bold),  
 ),  
 SizedBox(height: 20),  
 Text(  
 'Elapsed Time:',  
 style: TextStyle(fontSize: 16),  
 ),  
 Text(  
 \_formatElapsedTime(\_stopwatch.elapsed),  
 style: TextStyle(fontSize: 24, fontWeight: FontWeight.bold),  
 ),  
 SizedBox(height: 20),  
 Row(  
 mainAxisAlignment: MainAxisAlignment.center,  
 children: [  
 ElevatedButton(  
 onPressed: \_startCountingSteps,  
 child: Text('Start'),  
 ),  
 SizedBox(width: 20),  
 ElevatedButton(  
 onPressed: \_stopCountingSteps,  
 child: Text('Stop'),  
 ),  
 SizedBox(width: 20),  
 ElevatedButton(  
 onPressed: \_resetTime,  
 child: Text('Reset'),  
 ),  
 ],  
 ),  
 ],  
 ),  
 ),  
 );  
 }  
}

# Widgets

## widget\_utils

import 'package:flutter/material.dart';  
import 'package:flutter/services.dart';  
import 'package:flutter\_svg/svg.dart';  
import 'package:get/get.dart';  
import 'package:responsive\_sizer/responsive\_sizer.dart';  
import '../../utils/color\_data.dart';  
import '../../utils/constant.dart';  
  
Widget getVerSpace(double verSpace) {  
 return SizedBox(  
 height: verSpace,  
 );  
}  
  
Widget getHorSpace(double verSpace) {  
 return SizedBox(  
 width: verSpace,  
 );  
}  
  
Widget getAssetImage(String image,  
 {double? width,  
 double? height,  
 Color? color,  
 double? scale,  
 BoxFit boxFit = BoxFit.contain}) {  
 return Image.asset(  
 Constant.assetImagePath + image,  
 color: color,  
 width: width,  
 height: height,  
 fit: boxFit,  
 scale: scale,  
 );  
}  
  
Widget getSvgImage(String image,  
 {double? width,  
 double? height,  
 Color? color,  
 BoxFit boxFit = BoxFit.contain}) {  
 return SvgPicture.asset(  
 Constant.svgImagePath + image,  
 width: width,  
 height: height,  
 fit: boxFit,  
 );  
}  
  
getWhiteColorStatusBar() {  
 return AppBar(  
 backgroundColor: backgroundColor,  
 toolbarHeight: 0,  
 elevation: 0,  
 systemOverlayStyle: SystemUiOverlayStyle(  
 systemNavigationBarColor: backgroundColor,  
 statusBarColor: backgroundColor,  
 statusBarIconBrightness: Brightness.dark),  
 );  
}  
  
getAppBar({leading, actions, text}) {  
 return AppBar(  
 backgroundColor: accentColor,  
 toolbarHeight: 10.h,  
 elevation: 0,  
 centerTitle: true,  
 leading: leading,  
 actions: [Padding(padding: EdgeInsets.only(right: 2.3.h), child: actions)],  
 title: getCustomTextW6S20(text: text, color: backgroundColor),  
 );  
}  
  
Widget getCustomTextW6S12({text, color, textAlign}) {  
 return Text(  
 text,  
 textAlign: textAlign ?? TextAlign.left,  
 style: TextStyle(  
 color: color,  
 fontWeight: FontWeight.w600,  
 fontSize: 12.px,  
 fontFamily: Constant.fontsFamily,  
 ),  
 );  
}  
  
Widget getMartTitle({text, color, maxLines}) {  
 return Text(  
 text,  
 overflow: TextOverflow.ellipsis,  
 maxLines: maxLines ?? 1,  
 style: TextStyle(  
 color: color,  
 fontWeight: FontWeight.bold,  
 fontSize: 12.px,  
 fontFamily: Constant.fontsFamily,  
 ),  
 );  
}  
  
Widget getMartTitleDetails({text, color}) {  
 return Text(  
 text,  
 overflow: TextOverflow.ellipsis,  
 maxLines: 2,  
 style: TextStyle(  
 color: color,  
 fontWeight: FontWeight.bold,  
 fontSize: 16.px,  
 fontFamily: Constant.fontsFamily,  
 ),  
 );  
}  
  
Widget getCustomTextW6S9({text, color}) {  
 return Text(  
 text,  
 maxLines: 2,  
 overflow: TextOverflow.ellipsis,  
 style: TextStyle(  
 color: color,  
 fontWeight: FontWeight.w600,  
 fontSize: 9.px,  
 fontFamily: Constant.fontsFamily,  
 ),  
 );  
}  
  
Widget getCustomTextW6S10({text, color}) {  
 return Text(  
 text,  
 maxLines: 2,  
 overflow: TextOverflow.ellipsis,  
 style: TextStyle(  
 color: color,  
 fontWeight: FontWeight.w600,  
 fontSize: 10.px,  
 fontFamily: Constant.fontsFamily,  
 ),  
 );  
}  
  
Widget getCustomTextW6S15({text, color, textAlign,maxLines}) {  
 return Text(  
 text,  
 textAlign: textAlign,  
 maxLines: maxLines,  
 style: TextStyle(  
 overflow: TextOverflow.ellipsis,  
 color: color,  
 fontWeight: FontWeight.w600,  
 fontSize: 15.px,  
 fontFamily: Constant.fontsFamily,  
 ),  
 );  
}  
  
Widget getCustomTextW6S17({text, color, textAlign}) {  
 return Text(  
 text,  
 textAlign: textAlign,  
 style: TextStyle(  
 color: color,  
 fontWeight: FontWeight.w600,  
 fontSize: 17.px,  
 fontFamily: Constant.fontsFamily,  
 ),  
 );  
}  
  
Widget getCustomTextW6S20({text, color}) {  
 return Text(  
 text,  
 maxLines: 1,  
  
 style: TextStyle(  
 color: color,  
 overflow: TextOverflow.ellipsis,  
 fontWeight: FontWeight.w600,  
 fontSize: 20.px,  
 fontFamily: Constant.fontsFamily,  
 ),  
 );  
}  
  
Widget getCustomTextW6S16({text, color, textAlign,maxLines}) {  
 return Text(  
 text,  
 maxLines: maxLines,  
 textAlign: textAlign ?? TextAlign.center,  
 style: TextStyle(  
 color: color,  
 fontWeight: FontWeight.w600,  
 fontSize: 16.px,  
 fontFamily: Constant.fontsFamily,  
 ),  
 );  
}  
  
Widget getCustomTextW6S22({text, color, textAlign}) {  
 return Text(  
 text,  
 overflow: TextOverflow.ellipsis,  
 maxLines: 2,  
 textAlign: textAlign ?? TextAlign.center,  
 style: TextStyle(  
 color: color,  
 fontWeight: FontWeight.w600,  
 fontSize: 22.px,  
 fontFamily: Constant.fontsFamily,  
 ),  
 );  
}  
  
Widget getCustomTextW6S36({text, color}) {  
 return Text(  
 text,  
 style: TextStyle(  
 color: color,  
 fontWeight: FontWeight.w800,  
 fontSize: 36.px,  
 fontFamily: Constant.fontsFamily,  
 ),  
 );  
}  
Widget getCustomTextW6S30({text, color}) {  
 return Text(  
 text,  
 style: TextStyle(  
 color: color,  
 fontWeight: FontWeight.w800,  
 fontSize: 30.px,  
 fontFamily: Constant.fontsFamily,  
 ),  
 );  
}  
  
getCustomTextStyleW5S15({color}) {  
 return TextStyle(  
 color: color,  
 fontWeight: FontWeight.w500,  
 fontFamily: Constant.fontsFamily,  
 fontSize: 15.px,  
 );  
}  
  
getCustomTextStyleW4S13({color}) {  
 return TextStyle(  
 color: color,  
 fontWeight: FontWeight.w400,  
 fontFamily: Constant.fontsFamily,  
 fontSize: 13.px,  
 );  
}  
  
getCustomTextStyleW7S15({color}) {  
 return TextStyle(  
 color: color,  
 fontWeight: FontWeight.w700,  
 fontFamily: Constant.fontsFamily,  
 fontSize: 15.px,  
 );  
}  
  
getCustomTextStyleW10S13({color}) {  
 return TextStyle(  
 color: color,  
 fontWeight: FontWeight.bold,  
 fontFamily: Constant.fontsFamily,  
 fontSize: 13.px,  
 );  
}  
  
Widget getCustomTextFormField(  
 {controller,  
 validator,  
 hintText,  
 suffixIcon,  
 obscureText,  
 keyboardType,  
 prefixIcon,  
 onTap,  
 readOnly,  
 suffixIconConstraintsWidth,  
 onChanged}) {  
 return TextFormField(  
 keyboardType: keyboardType,  
 obscureText: obscureText ?? false,  
 cursorColor: textBlackColor,  
 controller: controller,  
 validator: validator,  
 onTap: onTap,  
 onChanged: onChanged,  
 readOnly: readOnly ?? false,  
 style: getCustomTextStyleW5S15(  
 color: textBlackColor,  
 ),  
 decoration: InputDecoration(  
 enabledBorder: OutlineInputBorder(  
 borderSide: BorderSide(color: backgroundColor, width: 1.0.px),  
 borderRadius: BorderRadius.circular(10.0.px)),  
 focusedBorder: OutlineInputBorder(  
 borderSide: BorderSide(color: backgroundColor, width: 1.0.px),  
 borderRadius: BorderRadius.circular(10.0.px)),  
 border: OutlineInputBorder(borderRadius: BorderRadius.circular(10.0.px)),  
 contentPadding: EdgeInsets.symmetric(vertical: 1.8.h, horizontal: 1.3.h),  
 suffixIconConstraints: BoxConstraints.tightFor(  
 width: suffixIconConstraintsWidth ?? 6.0.h, height: 2.5.h),  
 suffixIcon: suffixIcon,  
 prefixIconConstraints:  
 BoxConstraints.tightFor(width: 7.0.h, height: 3.5.h),  
 prefixIcon: prefixIcon,  
 isCollapsed: true,  
 filled: true,  
 fillColor: backgroundColor,  
 hintText: hintText,  
 hintStyle: getCustomTextStyleW5S15(  
 color: textBlackColor,  
 ),  
 enabled: true,  
 ),  
 );  
}  
  
Widget fillColorButton({text, color, paddingVertical}) {  
 return Container(  
 width: double.infinity,  
 alignment: Alignment.center,  
 padding: EdgeInsets.symmetric(vertical: paddingVertical ?? 1.2.h),  
 decoration: BoxDecoration(  
 color: color ?? accentColor,  
 borderRadius: BorderRadius.all(Radius.circular(10.px)),  
 border: Border.all(color: color ?? accentColor, width: 1.px)),  
 child: getCustomTextW6S16(  
 text: text,  
 color: textWhiteColor,  
 ),  
 );  
}  
  
Widget timerButton({text}) {  
 return Container(  
 width: double.infinity,  
 alignment: Alignment.center,  
 margin: EdgeInsets.symmetric(horizontal: 5.h),  
 padding: EdgeInsets.symmetric(vertical: 0.5),  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius: BorderRadius.all(Radius.circular(10.px)),  
 border: Border.all(color: backgroundColor, width: 1.px)),  
 child: getCustomTextW6S36(  
 text: text,  
 color: pinkAppColor,  
 ),  
 );  
}  
  
Widget outlineColorButton({text, color, borderColor}) {  
 return Container(  
 width: double.infinity,  
 alignment: Alignment.center,  
 padding: EdgeInsets.symmetric(vertical: 3.7.h),  
 decoration: BoxDecoration(  
 color: color,  
 borderRadius: BorderRadius.all(Radius.circular(18.px)),  
 border: Border.all(color: borderColor, width: 1.px)),  
 child: Row(  
 mainAxisAlignment: MainAxisAlignment.center,  
 crossAxisAlignment: CrossAxisAlignment.center,  
 children: [  
 getCustomTextW6S20(  
 text: text,  
 color: textWhiteColor,  
 ),  
 getHorSpace(2.h),  
 Icon(  
 Icons.arrow\_right\_outlined,  
 color: backgroundColor,  
 )  
 ],  
 ),  
 );  
}  
  
Widget outlineButton({text, textColor, color, borderColor, paddingVertical}) {  
 return Container(  
 width: double.infinity,  
 alignment: Alignment.center,  
 padding: EdgeInsets.symmetric(vertical: paddingVertical ?? 1.6.h),  
 decoration: BoxDecoration(  
 color: color,  
 borderRadius: BorderRadius.all(Radius.circular(10.px)),  
 border: Border.all(color: borderColor, width: 1.px)),  
 child: getCustomTextW6S16(  
 text: text,  
 color: textColor ?? borderColor,  
 ),  
 );  
}  
  
// class ExpentionCard extends StatelessWidget {  
// dynamic heading, subheading;  
//  
// ExpentionCard({this.heading, this.subheading});  
//  
// @override  
// Widget build(BuildContext context) {  
// return ExpandableNotifier(  
// child: Column(  
// children: <Widget>[  
// ScrollOnExpand(  
// scrollOnExpand: true,  
// scrollOnCollapse: false,  
// child: ExpandablePanel(  
// theme: const ExpandableThemeData(  
// headerAlignment: ExpandablePanelHeaderAlignment.center,  
// tapBodyToCollapse: true,  
// hasIcon: false,  
// tapHeaderToExpand: false,  
// crossFadePoint: 20),  
// header: getCustomTextW6S12(text: heading, color: subTextColor),  
// collapsed: Text(  
// subheading,  
// softWrap: true,  
// maxLines: 2,  
// overflow: TextOverflow.ellipsis,  
// style: TextStyle(  
// color: textBlackColor,  
// fontWeight: FontWeight.w600,  
// fontSize: 12.px,  
// fontFamily: Constant.fontsFamily,  
// ),  
// ),  
// expanded: Column(  
// crossAxisAlignment: CrossAxisAlignment.start,  
// children: <Widget>[  
// for (var \_ in Iterable.generate(1))  
// Text(  
// subheading,  
// softWrap: true,  
// overflow: TextOverflow.fade,  
// style: TextStyle(  
// color: textBlackColor,  
// fontWeight: FontWeight.w600,  
// fontSize: 12.px,  
// fontFamily: Constant.fontsFamily,  
// ),  
// ),  
// ],  
// ),  
// builder: (\_, collapsed, expanded) {  
// return Expandable(  
// collapsed: collapsed,  
// expanded: expanded,  
// theme: const ExpandableThemeData(fadeCurve: Curves.bounceInOut),  
// );  
// },  
// ),  
// ),  
// if (subheading.toString().length >= 60)  
// Align(  
// alignment: Alignment.centerRight,  
// child: Builder(  
// builder: (context) {  
// var controller =  
// ExpandableController.of(context, required: true);  
// return GestureDetector(  
// onTap: () {  
// controller.toggle();  
// },  
// child: Text(  
// controller!.expanded ? "Read Less".tr : "Read More".tr,  
// style: TextStyle(color: textBlackColor, fontSize: 10.px),  
// ),  
// );  
// },  
// ),  
// ),  
// ],  
// ));  
// }  
// }  
  
MaterialBanner showMaterialBanner(BuildContext context,  
 {title, subTitle, onTapToCartScreen}) {  
 return MaterialBanner(  
 content: getCustomTextW6S12(text: title, color: accentColor),  
 padding: EdgeInsets.symmetric(horizontal: 2.h),  
 backgroundColor: backCardColor,  
 actions: [  
 GestureDetector(  
 onTap: onTapToCartScreen,  
 child: getCustomTextW6S12(text: subTitle, color: skyBlueTextColor)),  
 ]);  
}  
  
Widget settingFirstCard({icon, title, subTitle, arrowColor}) {  
 return Container(  
 height: 7.h,  
 padding: EdgeInsets.symmetric(horizontal: 2.h),  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius: BorderRadius.all(Radius.circular(10.px)),  
 boxShadow: [  
 BoxShadow(  
 color: borderColor,  
 blurRadius: 2,  
 offset: const Offset(0, 0), // changes position of shadow  
 ),  
 ],  
 ),  
 child: Row(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 mainAxisAlignment: MainAxisAlignment.spaceBetween,  
 children: [  
 Row(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 mainAxisAlignment: MainAxisAlignment.start,  
 children: [  
 SizedBox(width: 5.h, child: getSvgImage(icon, height: 3.h)),  
 getHorSpace(1.h),  
 Column(  
 crossAxisAlignment: CrossAxisAlignment.start,  
 mainAxisAlignment: MainAxisAlignment.center,  
 children: [  
 getCustomTextW6S15(text: title, color: textBlackColor),  
 getCustomTextW6S12(text: subTitle, color: subTextColor),  
 ],  
 ),  
 ],  
 ),  
 Icon(Icons.arrow\_forward\_ios\_rounded,  
 color: arrowColor ?? skyBlueTextColor, size: 2.h),  
 ],  
 ),  
 );  
}  
Widget personalInfoCard({ title, subTitle}) {  
 return Container(  
 height: 7.h,  
 width: 100.w,  
 padding: EdgeInsets.symmetric(horizontal: 2.5.h),  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius: BorderRadius.all(Radius.circular(10.px)),  
 boxShadow: [  
 BoxShadow(  
 color: borderColor,  
 blurRadius: 2,  
 offset: const Offset(0, 0), // changes position of shadow  
 ),  
 ],  
 ),  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.start,  
 mainAxisAlignment: MainAxisAlignment.center,  
 children: [  
 getCustomTextW6S15(text: title, color: textBlackColor),  
 getCustomTextW6S12(text: subTitle, color: subTextColor),  
 ],  
 ),  
 );  
}  
  
Widget settingCard({icon, title, arrowColor}) {  
 return Container(  
 height: 7.h,  
 padding: EdgeInsets.symmetric(horizontal: 2.h),  
 decoration: BoxDecoration(  
 color: backgroundColor,  
 borderRadius: BorderRadius.all(Radius.circular(10.px)),  
 boxShadow: [  
 BoxShadow(  
 color: borderColor,  
 blurRadius: 2,  
 offset: const Offset(0, 0), // changes position of shadow  
 ),  
 ],  
 ),  
 child: Row(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 mainAxisAlignment: MainAxisAlignment.spaceBetween,  
 children: [  
 Row(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 mainAxisAlignment: MainAxisAlignment.start,  
 children: [  
 SizedBox(width: 5.h,child: getSvgImage(icon, height: 3.h)),  
 getHorSpace(1.h),  
 getCustomTextW6S15(text: title, color: subTextColor),  
 ],  
 ),  
 Icon(Icons.arrow\_forward\_ios\_rounded,  
 color: arrowColor ?? skyBlueTextColor, size: 2.h),  
 ],  
 ),  
 );  
}  
  
Widget signOutButton() {  
 return Container(  
 width: double.infinity,  
 alignment: Alignment.center,  
 padding: EdgeInsets.symmetric(vertical: 1.1.h),  
 decoration: BoxDecoration(  
 color: pinkAppColor,  
 borderRadius: BorderRadius.all(Radius.circular(10.px)),  
 border: Border.all(color: pinkAppColor, width: 1.px)),  
 child: Row(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 mainAxisAlignment: MainAxisAlignment.center,  
 children: [  
 getSvgImage('signout\_icon.svg', height: 4.h),  
 getHorSpace(2.h),  
 getCustomTextW6S16(text: 'Sign Out'.tr, color: textWhiteColor)  
 ],  
 ),  
 );  
}