import { Platform } from 'react-native';  
export const ip = 'http://localhost:9700';  
export const appApi = ip + '/app';  
export const IS\_DEV = \_\_DEV\_\_;  
export const IS\_IOS = Object.is(Platform.OS, 'ios');  
export const IS\_ANDROID = !IS\_IOS;  
import React, { useState, useEffect, useRef } from 'react';  
import { connect } from 'react-redux';  
import { bindActionCreators } from 'redux';  
import {  
 View,  
 Text,  
 StyleSheet,  
 NativeModules,  
 TouchableOpacity,  
 findNodeHandle,  
 Platform,  
} from 'react-native';  
import { Spinner } from 'native-base'  
import { AppRoute } from '../../navigator/AppRoutes';  
import { getVerifyToken, getVerifyResult } from '../../store/home/index';  
import Theme from '../../style/colors';  
import HeaderCommon from '../Component/HeaderCommon'  
function AuthenticationPage(props) {  
 function changeSelect(i) {  
 setSelectIndex(selectIndex => (selectIndex = i));  
 }  
 function personalVerify() {  
 props.getVerifyToken({ userId }, res => { //userInfo.id  
 if (res.code) {  
 showToast(res.message);  
 return;  
 }  
 let { verifyToken, bizId } = res.data;  
 if (Platform.OS === 'ios') {  
 let domId = findNodeHandle(AliyunVerify.current);  
 console.log('domId, verifyToken', domId, verifyToken,)  
 NativeModules.RealPersonAuth.addEvent(domId, verifyToken, (state, errorCode, message) => {  
 console.log('state, errorCode, message', state, errorCode, message)  
 if (state === 'RPStatePass') {  
 setLoading(true);  
 props.getVerifyResult({ bizId }, (res) => {  
 setLoading(false);  
 if(res.code === 0) {  
 NavigatorService.navigate(AppRoute.HOME);  
 } else {  
 showToast(res.message)  
 console.log('res.message\*\*\*\*\*: ',res.message)  
 }  
 });  
 } else if (state === 'RPStateFail') {  
 showToast("认证不通过");  
 } else if (state === 'RPStateNotVerify') {  
 showToast("未完成认证");  
 }  
 });  
 return;  
 }  
 NativeModules.AliyunVerify.show(verifyToken, ret => {  
 if (ret === 'success') {  
 setLoading(true);  
 props.getVerifyResult({ bizId }, (res) => {  
 setLoading(false);  
 if(res.code === 0) {  
 NavigatorService.navigate(AppRoute.HOME);  
 } else {  
 showToast(res.message)  
 console.log('res.message\*\*\*\*\*: ',res.message)  
 }  
 });  
 } else {  
 showToast('认证失败');  
 }  
 });  
 });  
 }  
 function loadStatus() {  
 console.log('loadStatus^^^^^^^^^^^^^');  
 setUserStatus('audit\_pending'); //修改认证成功返回 audit\_pending  
 }  
 function handlerVerify() {  
 if (selectIndex === 0) {  
 personalVerify();  
 } else if (selectIndex === 1) {  
 const { params } = props.route;  
 if (params && params.userId) {  
 NavigatorService.navigate(AppRoute.IDCARDVERTIFY, { userId: params.userId, refreshStatus: () => { loadStatus(); } });  
 } else {  
 NavigatorService.navigate(AppRoute.IDCARDVERTIFY, { refreshStatus: () => { loadStatus(); } });  
 }  
 } else {  
 const { params } = props.route;  
 if (params && params.userId) {  
 NavigatorService.navigate(AppRoute.PASSPORTVERTIFY, { userId: params.userId, refreshStatus: () => { loadStatus(); } });  
 } else {  
 NavigatorService.navigate(AppRoute.PASSPORTVERTIFY, { refreshStatus: () => { loadStatus(); } });  
 }  
 }  
 }  
 useEffect(() => {  
 setUser();  
 }, []); //[props.userInfo]  
 function setUser() {  
 const { params } = props.route;  
 if (params && params.userId) {  
 setUserId(params.userId);  
 setUserStatus(params.authStatus);  
 } else {  
 console.log('userId1:\*\*\*\*\*', props.userInfo.data.id)  
 setUserId(props.userInfo.data.id);  
 setUserStatus(props.userInfo.data.status);  
 }  
 }  
 const [AuthList] = useState([  
 {  
 name: '人脸识别',  
 type: '自动验证',  
 description: '适合中国公民，5分钟内验证完毕',  
 id: 1,  
 },  
 {  
 name: '身份证',  
 type: '人工审核',  
 description: '适合中国老人小孩或人脸识别失败的用户，两个工作日内完成',  
 id: 2,  
 },  
 {  
 name: '护照',  
 type: '人工审核',  
 description: '适合持有护照的非中国公民，两个工作日内完成',  
 id: 3,  
 },  
 ]);  
 const [selectIndex, setSelectIndex] = useState(0);  
 const [userId, setUserId] = useState('');  
 const [userStatus, setUserStatus] = useState('');  
 const [loading, setLoading] = useState(false);   
 const AliyunVerify = useRef();  
 return (  
 loading ? <Spinner style={STYLES.spinner} color="#5C8BFF"/> :  
 <View style={styles.container0}>  
 <HeaderCommon  
 options={{  
 backTitle: '返回',  
 title: '实名认证'  
 }}  
 />  
 <View style={styles.container} ref={AliyunVerify}>  
 <View>  
 <Text style={styles.tipStatus}>当前状态：</Text>  
 <Text style={styles.status}>{props.dictionaryMappings.user\_status[userStatus]}</Text>  
 </View>  
 <Text style={styles.verTitle}>请选择认证方式：</Text>  
 {AuthList.map((u, i) => {  
 return (  
 <TouchableOpacity style={i === selectIndex ? styles.selectedStyle : styles.authListStyle}  
 key={i}  
 onPress={() => changeSelect(i)}>  
 <Text style={i === selectIndex ? styles.SelectName : styles.name}>  
 {u.name}  
 </Text>  
 <Text style={i === selectIndex ? styles.selectTypeText : styles.typeText}>  
 {u.type}  
 </Text>  
 <Text style={i === selectIndex ? styles.selectSecondaryText : styles.secondaryText}>  
 {u.description}  
 </Text>  
 </TouchableOpacity>  
 );  
 })}  
 <Text style={styles.tipText}>  
 {"为提高识别成功率： \n1、请本人认证 \n2、拍照请保持环境光线适中 \n3、面部清晰可见无遮挡"}  
 </Text>  
 <TouchableOpacity style={styles.verifyBtn} onPress={handlerVerify}>  
 <Text style={styles.btnText}>开始认证</Text>  
 </TouchableOpacity>  
 </View>  
 </View>  
 );  
}  
function mapStateToProps(state) {  
 return {  
 userInfo: state.userInfo,  
 dictionaryMappings: state.dictionaryMappings  
 };  
}  
function matchDispatchToProps(dispatch) {  
 return bindActionCreators({ getVerifyToken, getVerifyResult }, dispatch);  
}  
export default connect(  
 mapStateToProps,  
 matchDispatchToProps,  
)(AuthenticationPage);  
const styles = StyleSheet.create({  
 container0: {  
 flex: 1,  
 },  
 container: {  
 flex: 1,  
 paddingHorizontal: 16,  
 paddingVertical: 16,  
 backgroundColor: Theme.background,  
 },  
 tipStatus: {  
 fontSize: $screen.scaleSize(14),  
 color: Theme.textSecondary,  
 },  
 status: {  
 position: 'absolute',  
 right: 0,  
 fontSize: $screen.scaleSize(14),  
 color: Theme.textDefault,  
 textAlign: 'right',  
 },  
 verTitle: {  
 marginVertical: 20,  
 fontSize: $screen.scaleSize(14),  
 color: Theme.textDefault,  
 },  
 authListStyle: {  
 marginBottom: 18,  
 padding: 16,  
 borderRadius: 4,  
 borderWidth: 1,  
 borderColor: Theme.border,  
 },  
 selectedStyle: {  
 marginBottom: 18,  
 padding: 16,  
 borderRadius: 4,  
 borderWidth: 1,  
 borderColor: '#527BDF',  
 },  
 SelectName: {  
 color: '#527BDF',  
 fontSize: $screen.scaleSize(18),  
 },  
 name: {  
 color: Theme.textDefault,  
 fontSize: $screen.scaleSize(18),  
 },  
 typeText: {  
 position: 'absolute',  
 top: 22,  
 right: 16,  
 fontSize: $screen.scaleSize(12),  
 color: Theme.textSecondary,  
 },  
 selectTypeText: {  
 position: 'absolute',  
 top: 22,  
 right: 16,  
 fontSize: $screen.scaleSize(12),  
 color: '#527BDF',  
 },  
 secondaryText: {  
 marginTop: 12,  
 fontSize: $screen.scaleSize(12),  
 color: Theme.textDefault,  
 },  
 selectSecondaryText: {  
 marginTop: 12,  
 fontSize: $screen.scaleSize(12),  
 color: '#527BDF',  
 },  
 tipText: {  
 padding: 12,  
 fontSize: $screen.scaleSize(12),  
 color: Theme.textSecondary,  
 borderRadius: 4,  
 lineHeight: 17,  
 backgroundColor: '#F0F0F0',  
 },  
 verifyBtn: {  
 position: 'absolute',  
 left: 16,  
 right: 16,  
 bottom: 50,  
 height: 40,  
 borderRadius: 20,  
 backgroundColor: '#5C8BFF'  
 },  
 btnText: {  
 height: 40,  
 lineHeight: 40,  
 textAlign: 'center',  
 fontSize: $screen.scaleSize(16),  
 color: '#FFFFFF',  
 },  
});  
import React, { useState, useEffect, useCallback, useRef } from 'react';  
import { connect } from 'react-redux';  
import { bindActionCreators } from 'redux';  
import { View, Text, StyleSheet, Animated, Platform, TouchableOpacity, FlatList } from 'react-native';  
import { getUserInfo, getMyHouseList } from '../../store/home/index';  
import { getAllData, getDictionaryMapping } from '../../store/login/index';  
import { getRecommandList } from '../../store/map/index';  
import { setHomeHouse, setCodeInfo, setDictionaryMappings } from '../../store/common/index'  
import { Root, Spinner } from 'native-base'  
import ActionSheet from 'react-native-custom-actionsheet'  
import { AppRoute } from '../../navigator/AppRoutes';  
import Swiper from '../Component/Swiper'  
import StatusCard from './Component/statusCard'  
import HouseItem from '../Component/housingList/item';  
import BlankPage from '../Component/BlankPage';  
import Theme from '../../style/colors';  
import { useFocusEffect } from '@react-navigation/native';  
import Geolocation from '@react-native-community/geolocation';  
import StickyHeader from 'react-native-stickyheader';  
import { STYLES } from '../../style/styles';  
const imgList = [ // 暂时写死  
 require('../../assets/images/mock/home1.jpg'),  
 require('../../assets/images/mock/home2.jpg'),  
 require('../../assets/images/mock/home3.jpg'),  
 require('../../assets/images/mock/home4.jpg')]  
function HomePage(props) {  
 const [scrollY] = useState(new Animated.Value(0))  
 const [headHeight] = useState(-1)  
 const [userInfo, setUserInfo] = useState({});  
 const [houseList, setHouseList] = useState({});  
 const [selectHouse, setSelectHouse] = useState({});  
 const [loadingStatus, setLoadingStatus] = useState(true)  
 const [loading, setLoading] = useState(true)  
 const ActionSheetRef = useRef(null);  
 const [ActionSheetConfig, setActionSheetConfig] = useState({  
 options: ['取消'],  
 TYPE: '',  
 CANCEL\_INDEX: 0  
 })  
 const [recommandList, setRecommandList] = useState(null);  
 const handleAppStateChange = () => {  
 props.getUserInfo((res) => {  
 if (res.code !== 0) {  
 NavigatorService.reset(AppRoute.LOGIN)  
 } else {  
 setLoading(false)  
 storageDataDictionary();  
 storageMappingDictionary();  
 initRemmcondList(); // 获取房源推荐  
 }  
 }); // 获取个人信息  
 }  
 useFocusEffect(  
 useCallback(() => {  
 handleAppStateChange()  
 }, [props.route])  
 )  
 const initRemmcondList = () => {  
 Geolocation.setRNConfiguration({  
 skipPermissionRequests: true  
 });  
 if (Platform.OS == 'ios') {  
 Geolocation.requestAuthorization()  
 }  
 console.log('get location start:')  
 Geolocation.getCurrentPosition(  
 position => {  
 console.log('position: ' + JSON.stringify(position))  
 if (position.coords) {  
 props.getRecommandList(position.coords, res => { // 获取推荐列表  
 setRecommandList(res.data)  
 })  
 }  
 },  
 error => {  
 props.getRecommandList({}, res => { // 获取推荐列表  
 setRecommandList(res.data)  
 })  
 }  
 )  
 }  
 useEffect(() => {  
 const Info = props.userInfo;  
 console.log('user', Info)  
 if (Info && !Info.code) {  
 storage.set('info', Info.data);  
 setUserInfo(Info.data)  
 if (Info.data && Info.data.status === 'audit\_pass') {  
 props.getMyHouseList() // 获取本人的房源  
 } else {  
 setLoadingStatus(false)  
 }  
 }  
 }, [props.userInfo]);  
 useEffect(() => {  
 if (props.myHouseList && props.myHouseList.data) {  
 let { data } = props.myHouseList  
 console.log('props', props.homeHouse);  
 setHouseList(data)  
 let result = data[0]  
 if (props.homeHouse) {  
 let res = data.filter((item) => item.houseId == props.homeHouse)  
 res.length && (result = res[0])  
 }  
 setSelectHouse(result)  
 }  
 setLoadingStatus(false)  
 }, [props.myHouseList])  
 const handleToDetailPage = item => {  
 NavigatorService.navigate(AppRoute.PUBLISHOUSEDETAIL, { id: item.id });  
 };  
 const renderContent = () => {  
 if (recommandList && recommandList.length) {  
 return (  
 <FlatList  
 data={recommandList}  
 keyExtractor={item => item.id}  
 renderItem={({ item, index }) => {  
 return (  
 <HouseItem  
 key={item.id}  
 houseInfo={item}  
 onPress={handleToDetailPage}  
 />  
 );  
 }}  
 />  
 )  
 } else {  
 return (<View style={{ top: 50 }}><BlankPage errorMsg='暂无房源' /></View>)  
 }  
 }  
 function storageDataDictionary() {  
 props.getAllData(res => {  
 let result = {};  
 res.data.forEach(item => {  
 result[item.name] = item.dictionaries;  
 });  
 props.setCodeInfo(result);  
 storage.set('code', result);  
 });  
 }  
 function storageMappingDictionary() {  
 props.getDictionaryMapping(res => {  
 props.setDictionaryMappings(res.data);  
 storage.set('dictionaryMappings', res.data);  
 });  
 }  
 const showList = () => {  
 let array = []  
 if (houseList.length) {  
 array = houseList.map((item) => {  
 item.text = item.address  
 return item  
 })  
 }  
 array.push({ text: "取消" })  
 const options = array.map((item) => {  
 return item.text  
 })  
 setActionSheetConfig({  
 CANCEL\_INDEX: array.length - 1,  
 options  
 })  
 setTimeout(() => {  
 ActionSheetRef.current.show()  
 })  
 }  
 return (  
 <Root>  
 {loading ? (<Spinner style={STYLES.spinner} color="#5C8BFF" />) :  
 <View style={styles.container}>  
 <Animated.ScrollView  
 bounces={false}  
 style={{ flex: 1 }}  
 onScroll={  
 Animated.event(  
 [{  
 nativeEvent: { contentOffset: { y: scrollY } } // 记录滑动距离  
 }],  
 { useNativeDriver: true }) // 使用原生动画驱动  
 }  
 scrollEventThrottle={1}  
 >  
 <View style={styles.header}>  
 <Text style={styles.title}>首页</Text>  
 <View style={styles.SwiperBox} >  
 <Swiper items={imgList} imgStyle={{ borderRadius: 10 }} />  
 </View>  
 {  
 loadingStatus ?  
 <Spinner style={STYLES.spinner} color="#5C8BFF" /> :  
 <StatusCard item={selectHouse} status={userInfo && userInfo.status} showList={() => showList()} />  
 }  
 </View>  
 {/\* <ActionSheet ref={(c) => { setActionSheet(c) }} /> \*/}  
 <ActionSheet  
 ref={ActionSheetRef}  
 options={ActionSheetConfig.options}  
 cancelButtonIndex={ActionSheetConfig.CANCEL\_INDEX}  
 onPress={(buttonIndex) => {  
 if (houseList[buttonIndex]) {  
 props.setHomeHouse(houseList[buttonIndex].houseId)  
 setSelectHouse(houseList[buttonIndex])  
 }  
 }}  
 />  
 <StickyHeader  
 style={{ backgroundColor: '#527BDF', padding: 0, }}  
 stickyHeaderY={headHeight} // 把头部高度传入  
 stickyScrollY={scrollY} // 把滑动距离传入  
 >  
 <View style={styles.listContent}>  
 <View style={{ flexDirection: 'row', justifyContent: 'space-between', alignItems: 'center' }}>  
 <Text style={styles.listTitle}>房源推荐</Text>  
 <TouchableOpacity style={styles.listMore} onPress={() => NavigatorService.navigate(AppRoute.HOUSELIST)}>  
 <Text style={{ color: Theme.textLink, fontSize: $screen.scaleSize(14) }} >查看更多</Text>  
 </TouchableOpacity>  
 </View>  
 </View>  
 </StickyHeader>  
 <View style={{ flex: 1 }}>  
 {renderContent()}  
 </View>  
 </Animated.ScrollView>  
 </View>  
 }  
 </Root>  
 );  
}  
function mapStateToProps(state) {  
 return {  
 userInfo: state.userInfo,  
 myHouseList: state.myHouseList,  
 webSocketInfo: state.webSocketInfo,  
 homeHouse: state.homeHouse  
 };  
}  
function matchDispatchToProps(dispatch) {  
 return bindActionCreators({ getUserInfo, getMyHouseList, getRecommandList, setHomeHouse, setDictionaryMappings, setCodeInfo, getDictionaryMapping, getAllData }, dispatch);  
}  
export default connect(  
 mapStateToProps,  
 matchDispatchToProps,  
)(HomePage);  
const styles = StyleSheet.create({  
 container: {  
 backgroundColor: '#fff',  
 flex: 1,  
 flexDirection: 'column'  
 },  
 header: {  
 backgroundColor: '#527BDF',  
 paddingHorizontal: 16,  
 paddingBottom: 16,  
 paddingTop: 8  
 },  
 title: {  
 color: '#fff',  
 marginBottom: 16,  
 fontSize: $screen.scaleSize(24)  
 },  
 SwiperBox: {  
 marginBottom: 16,  
 height: 150,  
 borderRadius: 10  
 },  
 listContent: {  
 backgroundColor: '#fff',  
 borderTopLeftRadius: 16,  
 borderTopRightRadius: 16,  
 padding: 16,  
 flex: 1,  
 flexDirection: 'column'  
 },  
 listTitle: {  
 fontSize: $screen.scaleSize(24),  
 color: '#282828'  
 },  
 listMore: {  
 }  
});  
import React, {useState, useEffect, useCallback} from 'react';  
import {connect} from 'react-redux';  
import {bindActionCreators} from 'redux';  
import {View, Text, StyleSheet} from 'react-native';  
import {getUserInfo} from '../../store/home/index';  
import {AppRoute} from '../../navigator/AppRoutes';  
import showToast from '../../util/toast';  
function AuditHouse(props) {  
 return (  
 <View style={styles.container}>  
 <Text style={styles.textFont}>  
 <Text>当前房源</Text>  
 <Text  
 style={styles.buttonTextStyle}  
 onPress={() => NavigatorService.navigate(AppRoute.HOUSEDETAIL)}>  
 审核中，  
 </Text>  
 <Text style={styles.textFont}>请耐心等待</Text>  
 </Text>  
 </View>  
 );  
}  
function mapStateToProps(state) {  
 return {  
 userInfo: state.userInfo,  
 };  
}  
function matchDispatchToProps(dispatch) {  
 return bindActionCreators({getUserInfo}, dispatch);  
}  
export default connect(  
 mapStateToProps,  
 matchDispatchToProps,  
)(AuditHouse);  
const styles = StyleSheet.create({  
 container: {  
 flex: 1,  
 justifyContent: 'center',  
 alignItems: 'center',  
 },  
 textFont: {  
 fontSize: $screen.scaleSize(20),  
 },  
 buttonTextStyle: {  
 color: Theme.primary,  
 },  
 secondaryText: {  
 fontSize: $screen.scaleSize(16),  
 marginTop: 10,  
 color: Theme.textSecondary,  
 },  
});  
import React, {useState, useEffect, useCallback} from 'react';  
import {connect} from 'react-redux';  
import {bindActionCreators} from 'redux';  
import {View, Text, Button, StyleSheet} from 'react-native';  
import {getUserInfo} from '../../store/home/index';  
import {AppRoute} from '../../navigator/AppRoutes';  
function HomePage(props) {  
 return (  
 <View style={styles.container}>  
 <View>  
 <Text style={styles.textFont}>您还没有添加任何门锁</Text>  
 <Text style={styles.secondaryText}>需要添加后才能执行开锁操作</Text>  
 <Button  
 title="登记房源"  
 onPress={() => NavigatorService.navigate(AppRoute.RECORD)}  
 />  
 <Button title="申请电子钥匙" />  
 </View>  
 </View>  
 );  
}  
function mapStateToProps(state) {  
 return {  
 userInfo: state.userInfo,  
 };  
}  
function matchDispatchToProps(dispatch) {  
 return bindActionCreators({getUserInfo}, dispatch);  
}  
export default connect(  
 mapStateToProps,  
 matchDispatchToProps,  
)(HomePage);  
const styles = StyleSheet.create({  
 container: {  
 flex: 1,  
 justifyContent: 'center',  
 alignItems: 'center',  
 },  
 textFont: {  
 fontSize: $screen.scaleSize(20),  
 },  
 buttonTextStyle: {  
 color: Theme.primary,  
 },  
 secondaryText: {  
 fontSize: $screen.scaleSize(16),  
 marginTop: 10,  
 color: Theme.textSecondary,  
 },  
});  
const vertifyCn = [  
 {  
 name: '真实姓名',  
 placeholder: '请输入真实姓名',  
 key: 'name',  
 type: 'INPUT',  
 required: true,  
 errorMsg: ['请输入真实姓名'],  
 },  
 {  
 name: '身份证号',  
 placeholder: '请输入身份证号',  
 key: 'identificationNo',  
 type: 'INPUT',  
 required: true,  
 errorMsg: ['请输入身份证号'],  
 },  
 {  
 name: '性别',  
 placeholder: '请选择性别',  
 key: 'gender',  
 type: 'RADIO',  
 initial: 'male',  
 required: true,  
 selectOptions: [{label: '男', value: 'male'}, {label: '女', value: 'female'}],  
 },  
 {  
 name: '民族',  
 placeholder: '请选择民族',  
 key: 'nation',  
 required: true,  
 type: 'PICKER', //'INPUT',   
 errorMsg: ['请选择民族'],  
 selectOptions: ["汉族", "蒙古族", "回族", "藏族", "维吾尔族", "苗族", "彝族", "壮族", "布依族", "朝鲜族", "满族", "侗族",  
 "瑶族", "白族", "土家族", "哈尼族", "哈萨克族", "傣族", "黎族", "傈僳族", "佤族", "畲族", "高山族", "拉祜族", "水族", "东乡族",  
 "纳西族", "景颇族", "柯尔克孜族", "土族", "达斡尔族", "仫佬族", "羌族", "布朗族", "撒拉族", "毛难族", "仡佬族", "锡伯族", "阿昌族",  
 "普米族", "塔吉克族", "怒族", "乌孜别克族", "俄罗斯族", "鄂温克族", "崩龙族", "保安族", "裕固族", "京族", "塔塔尔族", "独龙族",  
 "鄂伦春族", "赫哲族", "门巴族", "珞巴族", "基诺族"],  
 },  
 {  
 name: '出生日期',  
 required: true,  
 placeholder: '请选择出生日期',  
 key: 'birthDate',  
 type: 'DATE',  
 errorMsg: ['请选择出生日期'],  
 },  
 {  
 name: '身份证地址',  
 required: true,  
 placeholder: '请输入身份证地址',  
 key: 'identificationAddress',  
 type: 'INPUT',  
 errorMsg: ['请输入身份证地址'],  
 },  
];  
export default vertifyCn;  
const houseLayoutCn = [  
 {  
 name: '朝向',  
 key: 'direction',  
 headKey: 'houseLayout',  
 codeKey: 'house\_direction'  
 }, {  
 name: '面积',  
 key: 'area',  
 headKey: 'houseLayout'  
 }, {  
 name: '楼层',  
 key: ['floorCount', 'floor'],  
 keyType: 'array',  
 headKey: 'houseLayout'  
 }, {  
 name: '类型',  
 key: 'houseType',  
 codeKey: 'house\_type'  
 }  
]  
const houseItemCn = {  
 "air\_conditioner": "database",  
 "balcony": "dingding",  
 "gas": "twitter",  
 "gas\_stove": "aliyun",  
 "heater": "rocket1",  
 "induction\_cooker": "sound",  
 "microwave\_oven": "shake",  
 "refrigerator": "API",  
 "smoking\_machine": "android1",  
 "sofa": "indent-right",  
 "table\_chair": "disconnect",  
 "tv": "tablet1",  
 "wardrobe": "printer",  
 "washing\_machine": "Safety",  
 "water\_heater": "woman"  
}  
const houseRatePlanCn = [{  
 name: '水费',  
 key: 'waterFeeUnitPrice',  
 unit: '元/吨'  
}, {  
 name: '电费',  
 key: 'electricityFeeUnitPrice',  
 unit: '元/度'  
}, {  
 name: '宽带',  
 key: 'networkFeeUnitPrice',  
 unit: '元/月'  
}, {  
 name: '物业',  
 key: 'managementFeeUnitPrice',  
 unit: '元/平方米'  
}, {  
 name: '燃气',  
 key: 'gasFeeUnitPrice',  
 unit: '元/立方米'  
}, {  
 name: '暖气',  
 key: 'heatingFeeUnitPrice',  
 unit: '元/立方米'  
}]  
export {  
 houseLayoutCn,  
 houseItemCn,  
 houseRatePlanCn  
};  
const vertifyCn = [  
 {  
 name: '真实姓名',  
 placeholder: '请输入真实姓名',  
 key: 'name',  
 type: 'INPUT',  
 required: true,  
 errorMsg: ['请输入真实姓名'],  
 },  
 {  
 name: '护照号',  
 required: true,  
 placeholder: '请输入护照号',  
 key: 'identificationNo',  
 type: 'INPUT',  
 errorMsg: ['请输入护照号'],  
 },  
 {  
 name: '性别',  
 placeholder: '请选择性别',  
 key: 'gender',  
 type: 'RADIO',  
 initial: 'male',  
 selectOptions: [{label: '男', value: 'male'}, {label: '女', value: 'female'}],  
 },  
 {  
 name: '出生日期',  
 required: true,  
 placeholder: '请选择出生日期',  
 key: 'birthDate',  
 type: 'DATE',  
 errorMsg: ['请选择出生日期'],  
 },  
 {  
 name: '国籍',  
 required: true,  
 placeholder: '请输入国籍',  
 key: 'country',  
 type: 'INPUT',  
 errorMsg: ['请输入国籍'],  
 },  
];  
export default vertifyCn;  
import {connect} from 'react-redux';  
import {bindActionCreators} from 'redux';  
import { verifyIdCard } from '../../../store/home/index';  
import {getManualAuditInfo} from '../../../store/user/index';  
import Component from './component';  
function mapStateToProps(state) {  
 return {  
 userInfo: state.userInfo  
 };  
}  
function matchDispatchToProps(dispatch) {  
 return bindActionCreators({verifyIdCard, getManualAuditInfo}, dispatch);  
}  
export default connect(  
 mapStateToProps,  
 matchDispatchToProps  
)(Component);  
import React, { useState, useEffect } from 'react';  
import { View, Text, StyleSheet, ScrollView, TouchableOpacity } from 'react-native';  
import Form from '../../Component/form'  
import ImageUpload from '../../Component/imageUpload'  
import vertifyCn from '../config/PassportVertify'  
import { Spinner, Content } from 'native-base'  
import HeaderCommon from '../../Component/HeaderCommon'  
import { AppRoute } from '../../../navigator/AppRoutes';  
export default function PassportVertifyPage(props) {  
 const [formData, setFormData] = useState({});  
 const [formImages, setFormImages] = useState([]);  
 const [userId, setUserId] = useState(null);  
 const [oldData, setOldData] = useState({});  
 const [loading, setLoading] = useState(false);  
 useEffect(() => {  
 setUId();  
 getAuthInfo();  
 }, []); //[props.userInfo]  
 function setUId() {  
 const {params} = props.route;  
 if (params && params.userId) {  
 setUserId(params.userId);  
 } else {  
 setUserId(props.userInfo.data.id)  
 }  
 }  
 function getAuthInfo() {  
 var uId = props.userInfo.data.id;  
 const {params} = props.route;  
 if (params && params.userId) {  
 uId= params.userId;  
 }  
 setLoading(true);  
 props.getManualAuditInfo(uId, res => {  
 setLoading(false);  
 if (res.code === 0) {  
 if (!(res.data)  
 || (res.data.identificationType !== 'passport')) {   
 return;  
 }  
 let {name, identificationNo, gender, birthDate, country} = res.data;  
 var oldData = {name, identificationNo, gender, birthDate, country} ;  
 if (!oldData.gender) {  
 oldData.gender = 'male';  
 }  
 setOldData(Object.assign({}, oldData));  
 setFormData(oldData);  
 let imgs = [];  
 if (res.data.imageUrls && res.data.imageUrls.length) {  
 res.data.imageUrls.map((item, index) => {  
 $getImage(item, response => {  
 let data = Object.assign([], imgs);  
 data[index] = response;  
 setFormImages(data);  
 imgs[index] = response;  
 }, true)  
 })  
 }  
 } else {  
 showToast(res.message);  
 }  
 });  
 }  
 const handleConfirm = () => {  
 let message = ''  
 let key = vertifyCn.findIndex((item) => { return (item.required && !formData[item.key]) })  
 message = vertifyCn[key] && vertifyCn[key].errorMsg[0]  
 console.log('\*\*\*\*3:', formData);  
 if (formData) {  
 var nowDateTime = new Date();  
 var selDateTime = new Date(formData.birthDate);  
 if ( nowDateTime.getTime() < selDateTime.getTime()) {  
 message = '出生日期不能大于当前日期';  
 }  
 }   
 if (!message) {  
 for (var i = 0; i < 3; i++) {  
 let item = formImages[i]  
 if (!item) {  
 message = '请上传三张图片';  
 break;  
 }  
 }  
 }  
 if (message) {  
 showToast(message);  
 return;  
 }  
 var result = new FormData()  
 changeToForm(result)  
 result.append('userId', userId)  
 result.append('identificationType', 'passport')  
 setLoading(true);  
 console.log('88userId: ', userId);  
 console.log('huzhao^^^^^^^^: ',result);  
 props.verifyIdCard(result, (res) => {  
 console.log('huzhaorenzhxiugai: ',res)  
 setLoading(false);  
 if (res.code === 0) {  
 showToast('提交成功');  
 NavigatorService.navigate(AppRoute.VERDETAILS);  
 const {params} = props.route;  
 if (params && params.refreshStatus) {  
 params.refreshStatus();  
 }  
 } else {  
 if (res === "timeout" || res.message === "Network request failed") {  
 } else {  
 showToast(res.message)  
 }  
 }  
 })  
 };  
 const changeToForm = (result) => {  
 for (var index in formData) {  
 result.append(index, formData[index])  
 }  
 for (var i in formImages) {  
 result.append('images', formImages[i])  
 }  
 }  
 const changeForm = data => {  
 setFormData(data);  
 setOldData(data);  
 };  
 const setImageForm = (type, obj) => {  
 let data = Object.assign([], formImages)  
 data[type] = obj  
 setFormImages(data)  
 }  
 return (  
 loading ? <Spinner style={STYLES.spinner} color="#5C8BFF"/> :  
 <View style={styles.container}>  
 <HeaderCommon  
 options={{  
 backTitle: '返回',  
 title: '护照认证'  
 }}  
 />  
 <Content>  
 <ScrollView bounces={false} contentContainerStyle={styles.scrollContainer}>  
 <Text style={styles.textTitle}>基本资料</Text>  
 <Form   
 config={vertifyCn}   
 class={styles.formBox}   
 changeForm={changeForm}   
 oldData={oldData}  
 />  
 <Text style={styles.textTitle}>照片上传</Text>  
 <View style={styles.ImageUploadBox}>  
 <ImageUpload title='护照个人信息'   
 setImageForm={(obj) => setImageForm(0, obj)}   
 imgUrl={formImages[0] && formImages[0].uri}  
 />  
 <ImageUpload title='护照入境信息'   
 setImageForm={(obj) => setImageForm(1, obj)}   
 imgUrl={formImages[1] && formImages[1].uri}  
 />  
 <ImageUpload title='手持护照'   
 setImageForm={(obj) => setImageForm(2, obj)}   
 imgUrl={formImages[2] && formImages[2].uri}  
 />  
 </View>  
 <TouchableOpacity style={styles.Btn} onPress={() => { handleConfirm(); }}>  
 <Text style={styles.btnText}>确认</Text>  
 </TouchableOpacity>  
 </ScrollView>  
 </Content>  
 </View>  
 );  
}  
const styles = StyleSheet.create({  
 container: {  
 flex: 1,  
 },  
 scrollContainer: {  
 paddingTop: 16,  
 paddingHorizontal: 16,  
 backgroundColor: Theme.background,  
 },  
 textTitle: {  
 fontSize: $screen.scaleSize(16),  
 color: Theme.textDefault,  
 },  
 formBox: {  
 paddingBottom: 16,  
 marginTop: 10,  
 },  
 ImageUploadBox: {  
 flex: 1,  
 flexDirection: "row",  
 justifyContent: "space-between",  
 marginTop: 20,  
 },  
 Btn: {  
 marginTop: 100,  
 marginBottom: 50,  
 height: 40,  
 borderRadius: 20,  
 backgroundColor: Theme.primary,  
 },  
 btnText: {  
 height: 40,  
 lineHeight: 40,  
 textAlign: 'center',  
 fontSize: $screen.scaleSize(16),  
 color: '#FFFFFF',  
 },  
})import React, { useState, useEffect } from 'react';  
import { View, StyleSheet, Platform } from 'react-native'  
import { Button, Icon, Text, Left, Right, Header, Body, Input } from 'native-base';  
import Entypo from 'react-native-vector-icons/Entypo';  
import AntDesign from 'react-native-vector-icons/AntDesign';  
import { AppRoute } from '../../../navigator/AppRoutes';  
import EvilIcons from 'react-native-vector-icons/EvilIcons'  
export default function SearchHeader(props) {  
 const [searchItem, setSearchItem] = useState('');  
 return (  
 <View>  
 <Header hasSegment style={styles.container}>  
 {/\* <Left style={[{ flex: 1 }, Platform.OS == 'android' && {top: -2, left: -2}]}>  
 <Button transparent onPress={() => NavigatorService.goBack()} >  
 <AntDesign name="left" style={[styles.actionColor, Platform.OS == 'android' && {top: 2}]} />  
 <Text style={[styles.backBtn, Platform.OS == 'android' && styles.backBtnAndroid]}>返回</Text>  
 </Button>  
 </Left> \*/}  
 <Left style={[{ flex: 1 },]}>  
 <Button transparent onPress={() => NavigatorService.goBack()} >  
 <AntDesign name="left" style={[styles.actionColor,]} />  
 <Text style={[styles.backBtn,]}>返回</Text>  
 </Button>  
 </Left>  
 <Body style={{ flexDirection: 'row', flex: 4, alignItems: 'center', backgroundColor: '#E9E9E9', height: 30, borderRadius: 20, }}>  
 <EvilIcons name='search' style={styles.searchIcon} />  
 <Input  
 style={styles.Input}  
 value={searchItem}  
 onChangeText={setSearchItem}  
 onBlur={() => props.getSearchParams(searchItem)}  
 placeholder='搜索房源' />  
 </Body>  
 <Right style={{ flex: 0.5 }}>  
 <Entypo style={styles.RightIcon} name='location' onPress={() => NavigatorService.navigate(AppRoute.MAPHOUSE)} />  
 </Right>  
 </Header>  
 </View>  
 );  
}  
const styles = StyleSheet.create({  
 container: {  
 height: 44,  
 padding: 10,  
 backgroundColor: '#fff'  
 },  
 selectBox: {  
 borderTopWidth: 1,  
 borderBottomWidth: 1,  
 backgroundColor: '#FFFFFF',  
 borderBottomColor: '#E9E9E9',  
 borderTopColor: '#E9E9E9',  
 paddingVertical: 12,  
 paddingHorizontal: 10,  
 flexDirection: 'row'  
 },  
 actionColor: {  
 color: Theme.textLink,  
 fontSize: $screen.scaleSize(14),  
 marginRight: 10,  
 left: 3  
 },  
 searchIcon: {  
 color: '#7C7C7C',  
 fontSize: $screen.scaleSize(18),  
 paddingLeft: 10,  
 },  
 Input: {  
 paddingLeft: 10,  
 fontSize: $screen.scaleSize(14)  
 },  
 select: {  
 flex: 2,  
 flexDirection: 'row',  
 textAlign: 'center',  
 alignItems: 'center'  
 },  
 RightIcon: {  
 color: '#333333',  
 fontSize: $screen.scaleSize(19),  
 width: '100%',  
 textAlign: 'center',  
 paddingVertical: 5  
 },  
 text: {  
 color: '#282828',  
 paddingHorizontal: 10  
 },  
 backBtn: {  
 color: Theme.backLink,  
 fontSize: 14,  
 fontWeight: 'normal',  
 top: 0,  
 },  
 backBtnAndroid: {  
 left: -10,  
 top: 1  
 }  
});  
import React, { useState, useEffect, useCallback } from 'react';  
import { View, Text, StyleSheet, TouchableOpacity } from 'react-native';  
import Theme from '../../../style/colors';  
import AntDesign from 'react-native-vector-icons/AntDesign';  
import Icomoon from '../../../common/Icomoon';  
import Entypo from 'react-native-vector-icons/Entypo';  
import { AppRoute } from '../../../navigator/AppRoutes';  
import MaterialCommunityIcons from 'react-native-vector-icons/MaterialCommunityIcons';  
const TENANT = 'tenant' //租客  
const status\_cf = {  
 'not\_audit': {  
 title: '您还未进行实名认证, 请尽快验证!',  
 desc: '更多操作需要实名认证才可以进行',  
 iconName: 'renzheng',  
 btnDesc: '实名认证',  
 route: 'AUTHENTICATION'  
 },  
 'audit\_pending': {  
 title: '您的实名信息审核中, 请耐心等待!',  
 desc: '更多操作需要认证完成才可以进行',  
 iconName: 'renzheng',  
 btnDesc: '查看进度',  
 route: 'VERDETAILS'  
 },  
 'audit\_reject': {  
 title: '您的实名信息未通过, 请重新提交!',  
 desc: '更多操作需要认证完成才可以进行',  
 iconName: 'renzheng',  
 btnDesc: '重新提交',  
 route: 'VERDETAILS'  
 },  
 'audit\_pass': {  
 title: '您还没添加登记房源, 请尽快登记!',  
 desc: '添加后才能执行开锁操作',  
 iconName: 'dengjifangyuan',  
 btnDesc: '登记房源',  
 route: 'RECORD'  
 }  
}  
const houseStatus = {  
 'audit\_pending': {  
 title: '您的房源正在审核中, 请耐心等待!',  
 desc: '审核完成后才可以添加住户和发布房源',  
 iconName: 'renzheng',  
 btnDesc: '查看进度',  
 showLocation: true,  
 route: 'HOUSEDETAIL'  
 },  
 'audit\_reject': {  
 title: '您的房源审核失败了, 请重新提交!',  
 desc: '审核完成后才可以添加住户和发布房源',  
 iconName: 'renzheng',  
 btnDesc: '重新提交',  
 showLocation: true,  
 route: 'HOUSEDETAIL'  
 },  
 'audit\_pass': {  
 title: '您的房源还未绑定设备, 请尽快绑定!',  
 desc: '审核完成后才可以添加住户和发布房源',  
 iconName: 'shebei',  
 btnDesc: '绑定设备',  
 showLocation: true,  
 route: 'FEATURE'  
 }  
}  
export default function StatusCard(props) {  
 const [options, setOptions] = useState(status\_cf['not\_audit'])  
 useEffect(() => {  
 if (props.status !== 'audit\_pass'  
 || (props.status === 'audit\_pass' && (!props.item || !props.item.houseId))) {  
 let data = Object.assign({}, status\_cf[props.status || 'not\_audit'])  
 setOptions(data)  
 }  
 }, [props, props.status])  
 useEffect(() => {  
 if (props.status === 'audit\_pass' && props.item && props.item.houseId) {  
 let { status, address, houseId, houseRole } = props.item  
 let data = Object.assign({}, houseStatus[status || 'audit\_pending'])  
 data.name = address  
 data.id = houseId  
 data.houseRole = houseRole  
 setOptions(data)  
 }  
 }, [props, props.item])  
 const renderStatus = () => {  
 var list = [  
 { icon: 'paper-plane', name: 1, text: '开启' },  
 { icon: '500px-with-circle', name: 2, text: '开启' },  
 { icon: 'progress-one', name: 3, text: '75%' },  
 { icon: 'signal', name: 4, text: '开启' },  
 { icon: 'paper-plane', name: 5, text: '开启' }  
 ]  
 return list.map((item) => {  
 return (  
 <View key={item.name} style={styles.cardList}>  
 <Entypo style={styles.LeftIcon} name={item.icon} />  
 <Text style={styles.cardText}>{item.text}</Text>  
 </View>  
 )  
 })  
 }  
 const renderRent = () => {  
 return (  
 <View style={styles.rentBox}>  
 {renderStatus()}  
 <TouchableOpacity style={styles.rightBox}>  
 <MaterialCommunityIcons style={[styles.LeftIcon, styles.rightText]} name='door' />  
 <Text style={styles.rightText}>开门</Text>  
 </TouchableOpacity>  
 </View>)  
 }  
 return (  
 <View style={styles.container}>  
 {options && options.showLocation &&  
 <TouchableOpacity style={styles.topBox} onPress={() => props.showList()}>  
 <Entypo style={styles.LeftIcon} name='location-pin' />  
 <Text style={styles.location} numberOfLines={1}>{options.name || '--'}</Text>  
 <Icomoon name='xiajiantou\_shixin' style={styles.RightIcon} />  
 </TouchableOpacity>  
 }  
 {/\* 判断是房东还是租客 \*/}  
 {(options && options.houseRole !== TENANT) && <View style={styles.bottomBox}>  
 <View style={styles.Leftcontent}>  
 <Text style={styles.title} numberOfLines={1}>{options.title || '--'}</Text>  
 <Text style={styles.des} note>{options.desc || '--'}</Text>  
 </View>  
 <TouchableOpacity style={styles.RightContent} onPress={() => NavigatorService.navigate(AppRoute[options.route], {  
 id: options.id || undefined,  
 role: options.houseRole || undefined,  
 })}>  
 {/\* <AntDesign name={options.iconName} style={styles.iconBox} /> \*/}  
 <Icomoon name={options.iconName} style={styles.iconBox} />  
 <Text style={{ color: Theme.primary, fontSize: $screen.scaleSize(14)}}>{options.btnDesc || '--'}</Text>  
 </TouchableOpacity>  
 </View>}  
 {options.houseRole && options.houseRole === TENANT && renderRent()}  
 </View>  
 );  
}  
const styles = StyleSheet.create({  
 container: {  
 backgroundColor: '#fff',  
 padding: 16,  
 borderRadius: 8  
 },  
 topBox: {  
 borderBottomWidth: 1,  
 borderBottomColor: '#E9E9E9',  
 fontSize: $screen.scaleSize(16),  
 paddingBottom: 13,  
 flexDirection: 'row'  
 },  
 location: {  
 color: '#7C7C7C',  
 fontSize: $screen.scaleSize(16),  
 flexWrap: 'wrap',  
 width: '80%'  
 },  
 RightIcon: {  
 position: 'absolute',  
 right: 20,  
 top: 4,  
 color: '#7C7C7C',  
 height: '100%',  
 textAlignVertical: 'center',  
 },  
 LeftIcon: {  
 color: '#7C7C7C',  
 paddingRight: 5,  
 fontSize: $screen.scaleSize(17),  
 textAlign: 'center',  
 top: 3  
 },  
 bottomBox: {  
 flexDirection: 'row',  
 alignItems: 'center',  
 },  
 title: {  
 color: '#282828',  
 fontSize: $screen.scaleSize(16),  
 paddingBottom: 7  
 },  
 des: {  
 color: '#7C7C7C',  
 fontSize: $screen.scaleSize(12)  
 },  
 Leftcontent: {  
 width: '80%'  
 },  
 RightContent: {  
 width: '20%',  
 alignItems: 'center',  
 justifyContent: 'center'  
 },  
 iconBox: {  
 color: Theme.primary,  
 fontSize: $screen.scaleSize(20),  
 paddingTop: 10,  
 paddingBottom: 2  
 },  
 rentBox: {  
 flexDirection: 'row',  
 marginVertical: 10,  
 },  
 cardList: {  
 marginRight: 15,  
 textAlign: 'center',  
 },  
 cardText: {  
 color: '#7C7C7C',  
 marginTop: 5  
 },  
 rightBox: {  
 flexDirection: 'row',  
 borderLeftWidth: 1,  
 borderLeftColor: '#E9E9E9',  
 paddingLeft: 15,  
 position: 'absolute',  
 right: 0  
 },  
 rightText: {  
 color: '#5C8BFF',  
 fontSize: $screen.scaleSize(16),  
 paddingTop: 10,  
 padding: 5  
 }  
});  
import React, { useState, useRef, useEffect } from 'react';  
import { View, StyleSheet, Linking } from 'react-native';  
import { Icon, Button, Left, Header, Text, Spinner } from 'native-base'  
import AntDesign from 'react-native-vector-icons/AntDesign'  
import Icomoon from '../../../common/Icomoon';  
import Theme from '../../../style/colors';  
import LocationsMap from '../../../page/Component/map/locations';  
import { houseLayoutCn, houseItemCn, houseRatePlanCn } from '../config/houseDetailCn'  
import showToast from '../../../util/toast';  
import Entypo from 'react-native-vector-icons/Entypo';  
const loc = {  
 "name": "房子",  
 latitude: 39.904989,  
 longitude: 116.40,  
 "type": 0  
}  
function HouseDetail(props) {  
 const [options, setOptions] = useState(null)  
 const [loading, setLoading] = useState(true)  
 const [code] = useState(props.code)  
 useEffect(() => {  
 setOptions(props.data)  
 console.log('props.data', props.data)  
 if (props.data) {  
 setLoading(false)  
 }  
 }, [props, props.data])  
 const changeCollection = (type, value) => {  
 let res = Object.assign({}, options)  
 res[type] = value  
 setOptions(res)  
 props.setCollection && props.setCollection({  
 publishInfoId: options.id,  
 add: value  
 }, (res) => {  
 if (!res.code && value) {  
 showToast('已收藏')  
 } else if (!res.code && !value) {  
 showToast('已取消收藏')  
 } else {  
 showToast(res.message || '操作失败')  
 }  
 })  
 }  
 const callPhone = () => {  
 let phone = options.mobile || +86  
 const url = `tel:${phone}`;  
 Linking.canOpenURL(url)  
 .then(supported => {  
 if (!supported) {  
 return showToast(`您的设备不支持该功能，请手动拨打 ${phone}`);  
 }  
 return Linking.openURL(url);  
 })  
 .catch(err => { showToast(`出错了：${err}`, 1.5); console.log(err) });  
 }  
 const renderHouseLayout = () => {  
 let result = houseLayoutCn.map((item) => {  
 let data = item.headKey ? options[item.headKey] : options  
 let text = ''  
 if (item.keyType === 'array') {  
 text = (data[item.key[1]] || '--') + '/' + (data[item.key[0]] || '--') + '层'  
 } else {  
 text = (item.codeKey ? code[item.codeKey][data[item.key]] : data[item.key]) || '--'  
 }  
 return (  
 <View style={styles.houseLayoutModule}>  
 <Text style={styles.houseLayoutTitle}>  
 {text}  
 </Text>  
 <Text style={styles.houseLayoutDesc}>{item.name}</Text>  
 </View>)  
 })  
 return (result.length && <View style={styles.houseLayoutBox}>{result}</View>) || null  
 }  
 const renderHouseAddition = (type) => {  
 let key = 'rent\_requirements'  
 let check = type === 'spots'  
 if (check) {  
 key = 'house\_spots'  
 }  
 let result = options.houseAddition[type].map((item) => {  
 return (<View style={[styles.houseAdditionModule, !check && { backgroundColor: '#E9E9E9' }]}>  
 <Text style={styles.houseAdditionText}>{code[key][item] || '--'}</Text>  
 </View>)  
 })  
 return (result.length && <View style={styles.houseAdditionBox}>{result}</View>) || null  
 }  
 const renderHouseItem = () => {  
 let result = options.houseAmenity.items.map((val) => {  
 return (<View style={styles.houseItemModule} key={val}>  
 <AntDesign name={houseItemCn[val]} style={styles.houseLayoutIcon} />  
 <Text style={styles.houseLayoutDesc}>{code['house\_item'][val] || '--'}</Text>  
 </View>)  
 })  
 return (result.length && <View style={styles.houseItemBox}>{result}</View>) || null  
 }  
 const renderHouseRatePlan = () => {  
 let result = houseRatePlanCn.map((val) => {  
 return (  
 <View style={styles.houseRatePlanModule} key={val.key}>  
 <Text style={styles.houseRatePlanDesc}>{val.name}:</Text>  
 <Text style={styles.houseRatePlanText}>{options.houseRatePlan[val.key] || '--'} {val.unit}</Text>  
 </View>  
 )  
 })  
 result.unshift(  
 <View style={styles.houseRatePlanModule}>  
 <Text style={styles.houseRatePlanDesc}>装修:</Text>  
 <Text style={styles.houseRatePlanText}>{code['house\_decorator'][options.houseAmenity.decoration] || '--'}</Text>  
 </View>)  
 result.unshift(  
 <View style={styles.houseRatePlanModule}>  
 <Text style={styles.houseRatePlanDesc}>户型:</Text>  
 <Text style={styles.houseRatePlanText}>{options.houseLayout.roomCount}室{options.houseLayout.hallCount}厅{options.houseLayout.toiletCount}卫</Text>  
 </View>)  
 return (result.length && <View style={styles.houseRatePlanBox}>{result}</View>) || null  
 }  
 return (  
 <View style={styles.container}>  
 {loading ? <Spinner style={STYLES.spinner} color="#5C8BFF" /> :  
 <View>  
 <View style={styles.flexRow}>  
 <Text style={styles.title}>{options.houseRatePlan.rentPrice}</Text>  
 <Text style={styles.desc}>元/月</Text>  
 <Text style={styles.depPay}>押{options.houseRatePlan.deposit}付{options.houseRatePlan.payment}</Text>  
 <Icomoon style={[styles.rightBtn, options.collectedByMe && styles.activeBtn]}  
 onPress={() => { changeCollection('collectedByMe', !options.collectedByMe) }}  
 name={options.collectedByMe ? 'aixin\_shixin' : 'aixin'}>  
 </Icomoon>  
 </View>  
 <Text style={styles.secondDes}>{options.title || '--'}</Text>  
 {renderHouseLayout()}  
 {renderHouseAddition('spots')}  
 <Text style={styles.moduleTitle}>房源简介</Text>  
 {renderHouseItem()}  
 {options.houseAddition.description ? <Text style={{ marginBottom: 16, color: '#282828', fontSize: $screen.scaleSize(12) }}>{options.houseAddition.description}</Text> : null}  
 {renderHouseAddition('requirements')}  
 {renderHouseRatePlan()}  
 <Text style={styles.moduleTitle}>地理位置</Text>  
 <View style={styles.locBox}>  
 <Entypo style={styles.locIcon} name='location-pin' />  
 <Text style={styles.locText}>{options.address || loc.name}</Text>  
 </View>  
 <View style={styles.mapBox}>  
 <LocationsMap loc={{  
 latitude: options.latitude || loc.latitude,  
 longitude: options.longitude || loc.longitude,  
 name: options.address || loc.name  
 }} />  
 </View>  
 <Button full style={styles.btn} onPress={() => callPhone()}>  
 <Text>电话咨询</Text>  
 </Button>  
 </View>  
 }  
 </View>  
 );  
}  
const styles = StyleSheet.create({  
 container: {  
 backgroundColor: '#fff',  
 height: '100%',  
 top: -50,  
 borderRadius: 20,  
 padding: 16  
 },  
 title: {  
 color: Theme.primary,  
 fontSize: $screen.scaleSize(24),  
 paddingRight: 5  
 },  
 desc: {  
 fontSize: $screen.scaleSize(12),  
 top: 10,  
 color: Theme.primary  
 },  
 flexRow: {  
 textAlignVertical: 'bottom',  
 flexDirection: 'row'  
 },  
 rightBtn: {  
 position: 'absolute',  
 right: 16,  
 fontSize: $screen.scaleSize(20),  
 color: '#666',  
 },  
 activeBtn: {  
 color: '#E7263E'  
 },  
 secondDes: {  
 color: '#282828',  
 fontWeight: 'bold',  
 marginTop: 8,  
 marginBottom: 24,  
 fontSize: $screen.scaleSize(18)  
 },  
 moduleTitle: {  
 color: '#282828',  
 marginVertical: 8,  
 },  
 houseLayoutBox: {  
 paddingHorizontal: 20,  
 paddingVertical: 10,  
 flexDirection: 'row',  
 justifyContent: 'space-between'  
 },  
 houseLayoutModule: {  
 flexDirection: 'column',  
 alignItems: 'center',  
 marginBottom: 10  
 },  
 houseLayoutTitle: {  
 color: '#282828',  
 fontSize: $screen.scaleSize(18),  
 marginBottom: 8  
 },  
 houseLayoutDesc: {  
 fontSize: $screen.scaleSize(12),  
 color: '#7C7C7C'  
 },  
 houseAdditionBox: {  
 flexWrap: 'wrap',  
 flexDirection: 'row'  
 },  
 houseAdditionModule: {  
 backgroundColor: '#ECF2FF',  
 marginRight: 16,  
 borderRadius: 5,  
 marginBottom: 10  
 },  
 houseAdditionText: {  
 fontSize: $screen.scaleSize(14),  
 color: '#7C7C7C',  
 paddingHorizontal: 15,  
 paddingVertical: 6  
 },  
 houseItemBox: {  
 flexDirection: 'row',  
 paddingVertical: 20,  
 flexWrap: 'wrap',  
 },  
 houseLayoutIcon: {  
 fontSize: $screen.scaleSize(24),  
 color: '#666666',  
 paddingBottom: 5  
 },  
 houseItemModule: {  
 alignItems: 'flex-start',  
 alignItems: 'center',  
 marginHorizontal: 10,  
 width: 40,  
 marginBottom: 25  
 },  
 houseRatePlanBox: {  
 flexDirection: 'row',  
 flexWrap: 'wrap',  
 marginVertical: 15,  
 flexDirection: 'row'  
 },  
 houseRatePlanModule: {  
 flexDirection: 'row',  
 width: '100%',  
 marginBottom: 20  
 },  
 houseRatePlanDesc: {  
 color: '#7C7C7C',  
 fontSize: $screen.scaleSize(14),  
 paddingRight: 5  
 },  
 houseRatePlanText: {  
 color: '#282828',  
 position: 'absolute',  
 right: 0,  
 fontSize: $screen.scaleSize(14)  
 },  
 mapBox: {  
 width: '100%',  
 height: 200,  
 backgroundColor: 'red',  
 marginTop: 10  
 },  
 btn: {  
 backgroundColor: '#5C8BFF',  
 borderRadius: 50,  
 marginTop: 35  
 },  
 locBox: {  
 backgroundColor: '#F5F7F9',  
 padding: 9,  
 marginBottom: 12,  
 flexDirection: 'row',  
 },  
 locIcon: {  
 textAlignVertical: 'center',  
 paddingRight: 10,  
 fontSize: 15,  
 color: '#7C7C7C'  
 },  
 locText: {  
 flex: 1,  
 color: '#282828',  
 fontSize: 14  
 },  
 depPay: {  
 backgroundColor: '#F5F7F9',  
 borderRadius: 4,  
 color: '#7C7C7C',  
 paddingHorizontal: 10,  
 height: 20,  
 textAlignVertical: 'center',  
 marginLeft: 10,  
 fontSize: $screen.scaleSize(12),  
 }  
});  
export default HouseDetail;  
import {connect} from 'react-redux';  
import {bindActionCreators} from 'redux';  
import {getHousingList} from '../../../store/home/index';  
import Component from './component';  
function mapStateToProps(state) {  
 return {};  
}  
function matchDispatchToProps(dispatch) {  
 return bindActionCreators({getHousingList}, dispatch);  
}  
export default connect(  
 mapStateToProps,  
 matchDispatchToProps,  
)(Component);  
import React, { useState, useRef, useEffect, useCallback } from 'react';  
import { View, StyleSheet, Alert } from 'react-native';  
import Geolocation from '@react-native-community/geolocation';  
import HouseListComponent from '../../Component/housingList/list';  
import SearchHeader from '../Component/searchHeader'  
import { useFocusEffect } from '@react-navigation/native';  
import DropdownMenu from '../../Component/housingList/filter';  
import BlankPage from '../../Component/BlankPage';  
function HouseList(props) {  
 const data = [{ type: 'text', value: '位置', key: 'location', selected: false }, { type: 'text', value: '户型', key: 'houseType', selected: false }, { type: 'text', value: '租金', key: 'rent', selected: false }, { type: 'icon', value: '排序', key: 'filter', selected: false }];  
 const filterParamsList = [  
 [{ distance: null }, { distance: '1' }, { distance: '2' }, { distance: '3' }],  
 [],  
 [{ priceHigh: null, priceLow: null }, { priceHigh: 1000, priceLow: 0 }, { priceHigh: 2000, priceLow: 1000 }, { priceHigh: 3000, priceLow: 2000 }, { priceHigh: 4000, priceLow: 3000 }, { priceHigh: null, priceLow: 4000 }],  
 [{ orderBy: 'newest' }, { orderBy: 'price\_up' }, { orderBy: 'price\_down' }]  
 ]  
 const [houseList, setHouseList] = useState([])  
 const [params, setParams] = useState({})  
 const [center, setCenter] = useState('');  
 const [positioningIsOn, setPositioningIsOn] = useState(true)  
 const [loading, setLoading] = useState(false)  
 useFocusEffect(useCallback(() => {  
 fetchHouseList()  
 getGeolocation()  
 }, [props.route]))  
 const getGeolocation = () => {  
 Geolocation.setRNConfiguration({  
 skipPermissionRequests: true  
 });  
 console.log('get location start:')  
 Geolocation.getCurrentPosition(  
 position => {  
 console.log('position: ' + JSON.stringify(position))  
 setPositioningIsOn(true)  
 if (position.coords) {  
 const arr = [position.coords.longitude, position.coords.latitude];  
 setCenter(arr.join(','))  
 }  
 },  
 error => {  
 setPositioningIsOn(false)  
 }  
 )  
 }  
 const fetchHouseList = (pageNum = 1) => {  
 setLoading(true)  
 const data = { ...params, ...{ pageNum: pageNum } }  
 props.getHousingList(data, (res) => {  
 console.log('hose', res)  
 setHouseList(res.data)  
 setLoading(false)  
 });  
 }  
 const renderContent = () => {  
 if (houseList.list && houseList.list.length) {  
 return (  
 <HouseListComponent list={houseList} handlerHouseList={(page) => fetchHouseList(page)} />  
 )  
 } else {  
 return (<BlankPage errorMsg='没有房源，请尝试更换筛选条件' />)  
 }  
 }  
 const getFilter = (sec, row) => {  
 if (sec === 1) {  
 return  
 }  
 if (sec === 0) {  
 if (!positioningIsOn && row) {  
 let options = [  
 { text: '取消', onPress: () => console.log('Ask me later pressed') },  
 { text: '确定', onPress: () => console.log('Ask me later pressed') },  
 {  
 cancelable: true,  
 onDismiss: () =>  
 console.log(  
 'This alert was dismissed by tapping outside of the alert dialog.'  
 )  
 }  
 ]  
 if (Platform.OS === 'android') {  
 options.splice(1, 0, {})  
 }  
 Alert.alert('需要开启定位才能看到附近的房源', '', options);  
 return  
 }  
 const lip = Object.assign(params, { center: center })  
 setParams(lip)  
 }  
 const data = filterParamsList[sec][row]  
 const newData = Object.assign(params, data)  
 setParams(newData)  
 fetchHouseList(1)  
 }  
 const getSection = (arr1, arr2) => {  
 const houseParams = {  
 houseType: arr1,  
 roomCount: arr2  
 }  
 const newData = Object.assign(params, houseParams)  
 console.log('newData2', newData);  
 setParams(newData)  
 fetchHouseList(1)  
 }  
 const getSearchParams = (item) => {  
 const data = { keyword: item }  
 const newData = Object.assign(params, data)  
 console.log('newData1', newData);  
 setParams(newData)  
 fetchHouseList(1)  
 }  
 return (  
 <View style={styles.container}>  
 <SearchHeader getSearchParams={(data) => getSearchParams(data)} />  
 <DropdownMenu  
 bgColor={'white'}  
 tintColor={'#282828'}  
 activityTintColor={'#5C8BFF'}  
 handler={(selection, row) => getFilter(selection, row)}  
 multipleSection={(arr1, arr2) => getSection(arr1, arr2)}  
 data={data}  
 >  
 {renderContent()}  
 </DropdownMenu>  
 {/\* </View>)} \*/}  
 </View>  
 );  
}  
const styles = StyleSheet.create({  
 container: {  
 flex: 1  
 }  
});  
export default HouseList;  
import { connect } from 'react-redux';  
import { bindActionCreators } from 'redux';  
import { getSearchSummaries } from '../../../store/map/index';  
import { getHousingList } from '../../../store/home/index';  
import Component from './component';  
function mapStateToProps(state) {  
 return {};  
}  
function matchDispatchToProps(dispatch) {  
 return bindActionCreators({ getSearchSummaries, getHousingList }, dispatch);  
}  
export default connect(  
 mapStateToProps,  
 matchDispatchToProps  
)(Component);  
import React, { useState, useRef, useCallback } from 'react';  
import { View, Text, StyleSheet, Dimensions, Platform, Alert } from 'react-native';  
import { MapView } from "react-native-amap3d";  
import BottomSheet from 'reanimated-bottom-sheet'  
import Geolocation from '@react-native-community/geolocation';  
import { useFocusEffect } from '@react-navigation/native';  
import HouseListComponent from '../../Component/housingList/list';  
import DropdownMenu from '../../Component/housingList/filter';  
import HeaderCommon from '../../Component/HeaderCommon';  
let h = Dimensions.get('window').height  
const snapPoints = [0.6 \* h, 0.5 \* h, 0.4 \* h, 0.3 \* h, 0.2 \* h, 0.1 \* h, 0]  
function MapHouse(props) {  
 const data = [{ type: 'text', value: '位置', key: 'location' }, { type: 'text', value: '方式/户型', key: 'houseType' }, { type: 'text', value: '租金', key: 'rent' }];  
 const filterParamsList = [  
 [{ distance: null }, { distance: '1' }, { distance: '2' }, { distance: '3' }],  
 [],  
 [{ priceHigh: null, priceLow: null }, { priceHigh: 1000, priceLow: 0 }, { priceHigh: 2000, priceLow: 1000 }, { priceHigh: 3000, priceLow: 2000 }, { priceHigh: 4000, priceLow: 3000 }, { priceHigh: null, priceLow: 4000 }],  
 [{ orderBy: 'newest' }, { orderBy: 'price\_up' }, { orderBy: 'price\_down' }]  
 ]  
 const [snapIndex] = useState(snapPoints.length - 1)  
 const [selectItem, setSelectItem] = useState()  
 const [markers, setMarker] = useState([])  
 const [params, setParams] = useState({})  
 const [positioningIsOn, setPositioningIsOn] = useState(true)  
 const [center, setCenter] = useState({  
 latitude: 39.904989,  
 longitude: 116.40  
 })  
 const [houseList, setHouseList] = useState({})  
 const mapRef = useRef()  
 var [loc, setLoc] = useState({})  
 const bottomSheetRef = React.useRef(null)  
 useFocusEffect(useCallback(() => {  
 Geolocation.setRNConfiguration({  
 skipPermissionRequests: true  
 });  
 if (Platform.OS == 'ios') {  
 Geolocation.requestAuthorization()  
 }  
 Geolocation.getCurrentPosition(  
 position => {  
 console.log('position: ' + JSON.stringify(position))  
 setPositioningIsOn(true)  
 if (position.coords) {  
 setCenter(position.coords);  
 }  
 },  
 error => {  
 setPositioningIsOn(false)  
 }  
 )  
 }, [props.route])  
 )  
 const selectMarker = (item) => {  
 setSelectItem(item)  
 if (item) {  
 props.getHousingList({  
 formattedAddress: item.formattedAddress  
 }, res => {  
 setHouseList(res.data.list)  
 })  
 bottomSheetRef.current.snapTo(3)  
 } else {  
 bottomSheetRef.current.snapTo(snapIndex)  
 }  
 }  
 const searchMarker = (reg) => {  
 let data = {}  
 let location = reg || loc  
 if (location && location.region) {  
 let { latitude, longitude, latitudeDelta, longitudeDelta } = location.region  
 let bottom = latitude - latitudeDelta / 2  
 let left = longitude - longitudeDelta / 2  
 let right = longitude + longitudeDelta / 2  
 let top = latitude + latitudeDelta / 2  
 let bottomRight = right + ',' + bottom  
 let topLeft = left + ',' + top  
 data = { bottomRight, topLeft }  
 if (params && !params.center) {  
 let center = Object.values(location.center).reverse().join(',')  
 data.center = center  
 }  
 }  
 let result = Object.assign({}, data, params)  
 console.log('params', result)  
 props.getSearchSummaries(result, res => {  
 if (!res.code && res.data) {  
 console.log('result', res.data)  
 setMarker(res.data)  
 } else {  
 setMarker([])  
 }  
 })  
 }  
 const mapClick = () => {  
 selectMarker(null)  
 }  
 const getFilter = (sec, row) => {  
 if (sec === 1) {  
 return  
 }  
 if (sec === 0) {  
 if (!positioningIsOn && row) {  
 let options = [  
 { text: '取消', onPress: () => console.log('Ask me later pressed') },  
 { text: '确定', onPress: () => console.log('Ask me later pressed') },  
 {  
 cancelable: true,  
 onDismiss: () =>  
 console.log(  
 'This alert was dismissed by tapping outside of the alert dialog.'  
 )  
 }  
 ]  
 if (Platform.OS === 'android') {  
 options.splice(1, 0, {})  
 }  
 Alert.alert('需要开启定位才能看到附近的房源', '', options);  
 return  
 }  
 const arr = [center.longitude, center.latitude];  
 const lip = Object.assign(params, { center: arr.join(',') })  
 setParams(lip)  
 }  
 const data = filterParamsList[sec][row]  
 const newData = Object.assign(params, data)  
 setParams(newData)  
 searchMarker()  
 }  
 const getSection = (arr1, arr2) => {  
 const houseParams = {  
 houseType: arr1,  
 roomCount: arr2  
 }  
 const newData = Object.assign(params, houseParams)  
 setParams(newData)  
 searchMarker()  
 }  
 const renderMarker = () => {  
 var Dom = []  
 Dom = markers.map((item) => {  
 let { latitude, longitude } = item  
 let checkSelect = selectItem && selectItem.formattedAddress === item.formattedAddress  
 return (  
 <MapView.Marker  
 key={item.formattedAddress}  
 title={item.formattedAddress}  
 icon={() => {  
 return (<Text style={[styles.markerBox, checkSelect && styles.selectMarker]}>{item.formattedAddress}</Text>)  
 }}  
 onPress={() => selectMarker(item)}  
 coordinate={{  
 latitude, longitude  
 }}  
 />  
 )  
 })  
 return Dom  
 }  
 const renderHeader = () => {  
 return (<Text style={styles.header}>——</Text>)  
 }  
 const renderContent = () => {  
 if (!selectItem) return (<View style={styles.content} />)  
 return (  
 <View style={styles.content}>  
 <View style={styles.contentBox}>  
 <Text style={styles.contentTitle}>{selectItem.formattedAddress}</Text>  
 <Text style={styles.contentDes}>{'在租' + selectItem.houseCount + '套'}</Text>  
 </View>  
 <HouseListComponent list={houseList} getList={(obj) => { getList(obj) }} />  
 </View>)  
 }  
 return (  
 <View style={{ flex: 1 }}>  
 <HeaderCommon  
 options={{  
 backTitle: '返回',  
 title: '地图找房'  
 }}  
 />  
 <DropdownMenu  
 bgColor={'white'}  
 tintColor={'#282828'}  
 activityTintColor={'#5C8BFF'}  
 handler={(selection, row) => getFilter(selection, row)}  
 multipleSection={(arr1, arr2) => getSection(arr1, arr2)}  
 data={data}  
 >  
 <MapView style={{ flex: 1 }}  
 zoomLevel={10}  
 locationEnabled  
 rotateEnabled={false}  
 onClick={() => mapClick()}  
 showsCompass={false}  
 onStatusChangeComplete={(region) => {  
 setLoc(region)  
 searchMarker(region)  
 }}  
 center={center}  
 ref={mapRef}>  
 {renderMarker()}  
 </MapView>  
 </DropdownMenu>  
 <BottomSheet  
 ref={bottomSheetRef}  
 snapPoints={snapPoints}  
 initialSnap={snapIndex}  
 enabledGestureInteraction={true}  
 enabledContentTapInteraction={false} // 巨坑：false，children组件的事件才会生效  
 enabledContentGestureInteraction={false}  
 enabledHeaderGestureInteraction={true}  
 style={styles.BottomSheet}  
 renderHeader={() => renderHeader()}  
 renderContent={() => renderContent()}  
 />  
 </View>  
 );  
}  
const styles = StyleSheet.create({  
 BottomSheet: {  
 position: 'absolute',  
 zIndex: 10,  
 backgroundColor: 'red',  
 width: '100%'  
 },  
 content: {  
 height: '100%',  
 textAlign: 'center',  
 backgroundColor: '#fff'  
 },  
 header: {  
 backgroundColor: '#fff',  
 textAlign: 'center'  
 },  
 markerBox: {  
 color: '#fff',  
 backgroundColor: '#527BDF',  
 padding: 5,  
 borderRadius: 5,  
 fontSize: $screen.scaleSize(12)  
 },  
 selectMarker: {  
 backgroundColor: '#FD8D22'  
 },  
 contentBox: {  
 flexDirection: 'row',  
 paddingHorizontal: 15,  
 paddingBottom: 10  
 },  
 contentTitle: {  
 color: '#282828',  
 fontSize: $screen.scaleSize(20),  
 flexWrap: 'wrap',  
 flex: 1  
 },  
 contentDes: {  
 color: '#7C7C7C',  
 fontSize: $screen.scaleSize(12),  
 paddingHorizontal: 10,  
 top: 8,  
 alignItems: 'center'  
 }  
})  
export default MapHouse;  
import { connect } from 'react-redux';  
import { bindActionCreators } from 'redux';  
import { getPublishHouseDetail, publishSetCollection } from '../../../store/publish/index';  
import Component from './component';  
function mapStateToProps(state) {  
 return {  
 publishHouseDetail: state.publishHouseDetail,  
 dictionaryMappings: state.dictionaryMappings  
 };  
}  
function matchDispatchToProps(dispatch) {  
 return bindActionCreators({ getPublishHouseDetail, publishSetCollection }, dispatch);  
}  
export default connect(  
 mapStateToProps,  
 matchDispatchToProps  
)(Component);  
import React, { useState, useRef, useEffect } from 'react';  
import { View, StyleSheet, ImageBackground, TouchableOpacity } from 'react-native';  
import { Icon, Button, Left, Header, Text, Spinner } from 'native-base'  
import HouseDetail from '../Component/houseDetail'  
import { ScrollView } from 'react-native-gesture-handler';  
import Swiper from '../../Component/Swiper';  
import HeaderCommon from '../../Component/HeaderCommon';  
const image = require('../../../assets/images/mock/home1.jpg')  
function PublicHouseDetail(props) {  
 const [data, setData] = useState({})  
 const [loading, setLoading] = useState(true)  
 const [code, setCode] = useState(true)  
 const [img, setImg] = useState([])  
 useEffect(() => {  
 let { id } = props.route.params  
 props.getPublishHouseDetail(id, res => {  
 if (res && !res.code && res.data) {  
 setData(Object.assign({}, res.data))  
 setLoading(false)  
 let picture = res.data.houseAddition.images  
 if (picture && picture.length) {  
 let i = picture.map((item) => {  
 return { uri: item }  
 })  
 setImg(i)  
 } else {  
 setImg([image])  
 }  
 }  
 })  
 setCode(props.dictionaryMappings)  
 }, [])  
 const setCollection = (data, callback) => {  
 props.publishSetCollection(data, callback)  
 }  
 return (  
 <View style={styles.container}>  
 <HeaderCommon  
 options={{  
 backTitle: '返回',  
 title: '房源信息',  
 }}  
 />  
 <ScrollView bounces={false}>  
 <View style={{ height: 250, backgroundColor: '#E1E5EB' }} >  
 {loading ? <Spinner color="#5C8BFF"></Spinner> : <Swiper items={img} showPage />}  
 </View>  
 {  
 loading ? <Spinner color="#5C8BFF"></Spinner> : <HouseDetail data={data} code={code} setCollection={(data, callback) => setCollection(data, callback)} />  
 }  
 </ScrollView>  
 </View>  
 );  
}  
const styles = StyleSheet.create({  
 container: {  
 position: 'relative',  
 backgroundColor: '#fff',  
 flexDirection: 'column'  
 },  
 headerBox: {  
 height: 300  
 }  
});  
export default PublicHouseDetail;  
import {connect} from 'react-redux';  
import {bindActionCreators} from 'redux';  
import { getManualAuditInfo, deleteTenant } from '../../../store/user/index';  
import Component from './component';  
function mapStateToProps(state) {  
 return {  
 userInfo: state.userInfo,  
 dictionaryMappings: state.dictionaryMappings  
 };  
}  
function matchDispatchToProps(dispatch) {  
 return bindActionCreators({getManualAuditInfo, deleteTenant}, dispatch);  
}  
export default connect(  
 mapStateToProps,  
 matchDispatchToProps  
)(Component);import React, {useState, useEffect} from 'react';  
import {View, Text, Image, TouchableOpacity, Alert, StyleSheet} from 'react-native';  
import { useFocusEffect } from '@react-navigation/native';  
import {Spinner} from 'native-base';  
import HeaderCommon from '../../Component/HeaderCommon'  
import Theme from '../../../style/colors';  
import {AppRoute} from '../../../navigator/AppRoutes';   
import { ScrollView } from 'react-native-gesture-handler';  
const auth\_status = {  
 'audit\_reject': {  
 title: '实名认证审核失败',  
 reasonDesc: '证件与本人不符，请修改',  
 btnTitle: '重新提交',  
 },  
 'audit\_pending': {  
 title: '实名认证审核中',  
 reasonDesc: '约两个工作日内完成审核',  
 btnTitle: '修改',  
 },  
 null: {  
 title: '服务返回认证状态null',  
 reasonDesc: '',  
 btnTitle: '',  
 },  
 'audit\_pass': {  
 title: '实名认证通过',  
 reasonDesc: '与上级界面状态可能不一致，数据有误',  
 btnTitle: '',  
 },  
}  
export default function VertifyDetailsPage(props) {  
 useFocusEffect(  
 React.useCallback(() => {  
 getAuthInfo();   
 }, [])  
 );  
 function getAuthInfo() {  
 var info = props.userInfo;  
 userId = info.data.id;  
 const {params} = props.route;  
 if (params && params.userId && userId !== params.userId) {  
 userId = params.userId;  
 setCanDelete(true);  
 } else {  
 setCanDelete(false);   
 }  
 props.getManualAuditInfo(userId, res => {  
 console.log('\*\*\*\*\*\*\*AuditInfo-kk:\*\*\*\*\*\*\*', res);  
 setLoading(false);  
 if (!res.code) {  
 dealDataRefresh(res.data);  
 } else {  
 showToast(res.message);  
 }  
 });  
 }  
 function dealDataRefresh(data) {   
 var authStatus = data.auditStatus;   
 if (authStatus === 'audit\_reject' && data.auditOpinions) {  
 auth\_status['audit\_reject'].reasonDesc = data.auditOpinions;  
 }  
 var sex = props.dictionaryMappings.gender[data.gender];  
 let actualDataArr;  
 if (data.identificationType === 'id\_card') {   
 actualDataArr = [   
 {title: "真实姓名", content: data.name},  
 {title: "身份证号码", content: data.identificationNo},  
 {title: "性别", content: sex},  
 {title: "民族", content: data.nation},  
 {title: "出生日期", content: data.birthDate},  
 {title: "身份证地址", content: data.identificationAddress},   
 ];  
 } else {  
 actualDataArr = [  
 {title: "真实姓名", content: data.name},  
 {title: "护照号", content: data.identificationNo},  
 {title: "性别", content: sex},  
 {title: "出生日期", content: data.birthDate},  
 {title: "国籍", content: data.country},   
 ];  
 }  
 setAuthStatus(authStatus);  
 setAuthOptions(auth\_status[authStatus]);  
 setData(actualDataArr);  
 console.log('data.imageUrls\*\*\*', data.imageUrls);  
 if (data.imageUrls && data.imageUrls[0]) {  
 $getImage(data.imageUrls[0], res => {  
 console.log(':&&&&&&&res: ', res);  
 setImage1({uri: res.uri});  
 }, true);  
 }  
 if (data.imageUrls && data.imageUrls[1]) {  
 $getImage(data.imageUrls[1], res => {  
 setImage2({uri: res.uri});  
 }, true);  
 }  
 if (data.imageUrls && data.imageUrls[2]) {  
 $getImage(data.imageUrls[2], res => {  
 setImage3({uri: res.uri});  
 }, true);  
 }  
 }  
 function handleModify() {  
 NavigatorService.navigate(AppRoute.AUTHENTICATION, {userId: userId, authStatus: authStatus});  
 }  
 function alertDelete() {  
 let options = [  
 { text: '取消', onPress: () => console.log('Ask me later pressed')},  
 { text: '确定', onPress: () => deleteTenant() },  
 {  
 cancelable: true,  
 onDismiss: () =>  
 console.log(  
 'This alert was dismissed by tapping outside of the alert dialog.'  
 )  
 }  
 ]  
 if (Platform.OS === 'android') {  
 options.splice(1, 0, {})  
 }  
 Alert.alert('确定删除', '', options);  
 }  
 function deleteTenant() {  
 const {params} = props.route;  
 console.log("%%%%2222222222", params);  
 const data = {  
 userId: params.userId,  
 houseId: params.houseId,  
 tenantUserId: params.tenantUserId,  
 };  
 props.deleteTenant(data, res => {  
 console.log('deleteTenant\*\*\*\*kkkk:', res);  
 if (!res.code) {  
 showToast("删除成功");  
 if (params.familyMember) {  
 NavigatorService.goBack();  
 } else {  
 NavigatorService.navigate(AppRoute.HOUSEDETAIL);  
 }  
 } else {  
 showToast(res.message);  
 }  
 });  
 }  
 function goBackOrTop() {  
 const {params} = props.route;  
 if (params && params.userId) {  
 NavigatorService.goBack();  
 } else {  
 props.navigation.popToTop();  
 }  
 }  
 const img = require('../../../assets/images/head.png');  
 const [authStatus, setAuthStatus] = useState('audit\_pending');  
 const [authOptions, setAuthOptions] = useState(auth\_status['audit\_pending']);   
 const [data, setData] = useState([]);   
 const [loading, setLoading] = useState(true);  
 const [image1, setImage1] = useState(img);   
 const [image2, setImage2] = useState(img);   
 const [image3, setImage3] = useState(img);   
 const [canDelete, setCanDelete] = useState(false);   
 return (  
 <View style={{flex: 1}}>  
 <HeaderCommon  
 options={{  
 backTitle: '返回',  
 title: '实名详情',  
 rightShow: 'flex',  
 rightTitle: authOptions.btnTitle,  
 leftPress: goBackOrTop,  
 rightPress: handleModify  
 }}  
 />  
 {loading ? <Spinner style={STYLES.spinner} color="#5C8BFF"/> :  
 <ScrollView bounces={false} contentContainerStyle={styles.containerStyle}>   
 <View style={[styles.topViewFail, authStatus === 'audit\_pending' && styles.topViewWait]}>  
 <Text style={styles.topTextStyle1}>{authOptions.title}</Text>  
 <Text style={styles.topTextStyle2}>{authOptions.reasonDesc}</Text>  
 </View>   
 {  
 data.map((item, index) => {   
 return (  
 <View>  
 <Text style={styles.textTitle}>{item.title}</Text>  
 <Text style={styles.textContent} numberOfLines={2} ellipsizeMode={'tail'}>{item.content}</Text>  
 </View>  
 );   
 })  
 }  
 <Text style={[styles.topTextStyle1, styles.space]}>{'照片上传'}</Text>   
 <View style={styles.ImageBox}>  
 <Image style={styles.imageStyle} source={image1}/>   
 <Image style={styles.imageStyle} source={image2}/>   
 <Image style={styles.imageStyle} source={image3} />   
 </View>  
 {!canDelete ? null :  
 <TouchableOpacity style={styles.btnStyle} onPress={alertDelete}>  
 <Text style={styles.btnTextStyle}>删除</Text>  
 </TouchableOpacity>  
 }  
 </ScrollView>   
 }   
 </View>  
 );  
}  
const styles = StyleSheet.create({   
 containerStyle: {  
 paddingHorizontal: 16,  
 paddingTop: 16,  
 backgroundColor: Theme.background,  
 },  
 topViewFail: {  
 padding: 12,  
 borderRadius: 4,  
 marginBottom: 10,  
 backgroundColor: '#FFECEC',  
 },  
 topViewWait: {  
 backgroundColor: '#ECF2FF',  
 },  
 topTextStyle1: {  
 fontSize: $screen.scaleSize(16),  
 color: Theme.textDefault,  
 },  
 topTextStyle2: {  
 fontSize: $screen.scaleSize(14),  
 color: Theme.textSecondary,  
 marginTop: 10,  
 },  
 textTitle: {  
 paddingVertical: 10,  
 fontSize: $screen.scaleSize(14),  
 color: Theme.textSecondary,  
 },  
 textContent: {  
 position: 'absolute',  
 left: 80,  
 right: 0,  
 paddingVertical: 10,  
 textAlign: 'right',  
 fontSize: $screen.scaleSize(14),  
 color: Theme.textDefault,  
 },  
 space: {  
 marginTop: 45,  
 },  
 ImageBox: {  
 flexDirection: "row",  
 justifyContent: "space-between",  
 marginTop: 20,  
 },  
 imageStyle: {  
 width: 116,  
 height: 87,  
 resizeMode: 'cover',  
 backgroundColor: '#f4f4f4',  
 borderWidth: 1,  
 borderColor: Theme.border,  
 borderRadius: 4,  
 },  
 btnStyle: {  
 marginTop: 100,  
 marginBottom: 50,  
 height: 40,  
 borderRadius: 20,  
 backgroundColor: '#E7263E'  
 },  
 btnTextStyle: {  
 height: 40,  
 lineHeight: 40,  
 textAlign: 'center',  
 fontSize: $screen.scaleSize(16),  
 color: '#FFFFFF',  
 },  
});  
import React, {useState, useEffect} from 'react';  
import { View, Text, Image, TouchableOpacity, Alert, StyleSheet } from 'react-native';  
import { Spinner } from 'native-base'  
import Theme from '../../../style/colors';  
import HeaderCommon from '../../Component/HeaderCommon'  
import {AppRoute} from '../../../navigator/AppRoutes';   
import {connect} from 'react-redux';  
import {bindActionCreators} from 'redux';  
import {getUserInfoById, deleteTenant} from '../../../store/user/index';  
const UserPassedPage = (props) => {  
 useEffect(() => {  
 getPassedUserInfo();   
 }, []);  
 function getPassedUserInfo() {  
 let userMeId = props.userInfo.data.id;   
 const {params} = props.route;  
 if (params.userId === userMeId) {  
 } else {  
 setCanDelete(true);  
 }  
 props.getUserInfoById(params.userId, res => {  
 console.log('\*\*\*\*getUserInfoById\*\*\*\*\*\*:', res);  
 setLoading(false);  
 if (!res.code) {  
 dealDataRefresh(res.data);  
 } else {  
 showToast(res.message);  
 }  
 });  
 }  
 function dealDataRefresh(data) {   
 let actualHeadData = {  
 headImage: {uri: data.avatarImageUrl},   
 name: data.name,  
 mobile: data.mobile,   
 };  
 if (!data.avatarImageUrl) {   
 const img = require('../../../assets/images/head.png');  
 actualHeadData.headImage = img;   
 }  
 setHeadData(actualHeadData);  
 var sex = props.dictionaryMappings.gender[data.gender];  
 var strNum = data.identificationNo;  
 if (strNum && strNum.length > 5) {  
 strNum = strNum.substr(0, 3) + "\*\*\*\*\*\*\*\*\*\*\*\*\*" + strNum.substr(strNum.length-2, 2)  
 }  
 const actualBasicData = [  
 {title: "性别", content: sex},  
 {title: "民族", content: data.nation},  
 {title: "出生日期", content: data.birthDate},  
 {title: "证件号", content: strNum},  
 {title: "所在区域", content: data.regionId},  
 {title: "详细地址", content: data.address},  
 {title: "教育程度", content: data.educationLevel},  
 ];  
 setBasicData(actualBasicData);  
 }  
 function alertDelete() {  
 let options = [  
 { text: '取消', onPress: () => console.log('Ask me later pressed')},  
 { text: '确定', onPress: () => deleteTenant() },  
 {  
 cancelable: true,  
 onDismiss: () =>  
 console.log(  
 'This alert was dismissed by tapping outside of the alert dialog.'  
 )  
 }  
 ]  
 if (Platform.OS === 'android') {  
 options.splice(1, 0, {})  
 }  
 Alert.alert('确定删除', '', options);  
 }  
 function deleteTenant() {  
 const {params} = props.route;  
 console.log("@@@@@@@@", params);  
 const data = {  
 userId: params.userId,  
 houseId: params.houseId,  
 tenantUserId: params.tenantUserId,  
 };  
 props.deleteTenant(data, res => {  
 console.log('deleteTenant\*\*\*\*kkkk:', res);  
 if (!res.code) {  
 showToast("删除成功");  
 if (params.familyMember) {  
 NavigatorService.goBack();  
 } else {  
 NavigatorService.navigate(AppRoute.HOUSEDETAIL);  
 }  
 } else {  
 showToast(res.message);  
 }  
 });  
 }  
 const [headData, setHeadData] = useState({});  
 const [basicData, setBasicData] = useState([]);  
 const [loading, setLoading] = useState(true);  
 const [canDelete, setCanDelete] = useState(true);   
 return (  
 <View style={styles.containerStyle0}>  
 <HeaderCommon  
 options={{  
 backTitle: '返回',  
 title: '成员详情'  
 }}  
 />  
 {loading ? <Spinner style={STYLES.spinner} color="#5C8BFF"/> :  
 <View style={styles.containerStyle}>  
 <Image style={styles.headImageStyle} source={headData.headImage} />  
 <Text style={styles.textName}>{headData.name}</Text>  
 <Text style={styles.textMobile}>{headData.mobile}</Text>  
 {   
 basicData.map((item, index) => {   
 return (  
 <View style={styles.sigContainer}>  
 <Text style={styles.textTitle}>{item.title}</Text>  
 <Text style={styles.textContent}>{item.content}</Text>  
 </View>  
 );   
 })  
 }  
 {!canDelete ? null :  
 <TouchableOpacity style={styles.btnStyle} onPress={alertDelete}>   
 <Text style={styles.btnTextStyle}>删除</Text>  
 </TouchableOpacity>  
 }  
 </View>  
 }  
 </View>  
 );   
}  
const styles = StyleSheet.create({   
 containerStyle0: {  
 flex: 1,  
 },  
 containerStyle: {  
 flex: 1,  
 paddingHorizontal: 16,  
 backgroundColor: Theme.background,  
 },  
 headImageStyle: {  
 marginTop: 29,  
 marginBottom: 17,  
 height: 48,   
 width: 48,  
 borderRadius: 24,  
 },  
 textName: {  
 position: 'absolute',  
 left: 88,  
 top: 31,   
 fontSize: $screen.scaleSize(14),  
 color: Theme.textDefault,  
 },  
 textMobile: {  
 position: 'absolute',  
 left: 88,  
 top: 59,   
 fontSize: $screen.scaleSize(14),  
 color: Theme.textDefault,  
 },  
 sigContainer: {  
 paddingVertical: 10,  
 },  
 textTitle: {  
 fontSize: $screen.scaleSize(14),  
 color: Theme.textSecondary,  
 },  
 textContent: {  
 position: 'absolute',  
 left: 80,  
 right: 0,  
 top: 10,  
 textAlign: 'right',  
 fontSize: $screen.scaleSize(14),  
 color: Theme.textDefault,  
 },  
 btnStyle: {  
 position: 'absolute',  
 left: 16,  
 right: 16,  
 bottom: 50,  
 height: 40,  
 borderRadius: 20,  
 backgroundColor: '#E7263E'  
 },  
 btnTextStyle: {  
 height: 40,  
 lineHeight: 40,  
 textAlign: 'center',  
 fontSize: $screen.scaleSize(16),   
 color: '#FFFFFF',   
 },  
});  
function mapStateToProps(state) {  
 return {  
 userInfo: state.userInfo,  
 dictionaryMappings: state.dictionaryMappings  
 };  
}  
function matchDispatchToProps(dispatch) {  
 return bindActionCreators({getUserInfoById, deleteTenant}, dispatch);   
}  
const VUserPassedPage = connect(  
 mapStateToProps,  
 matchDispatchToProps  
)(UserPassedPage);  
export default VUserPassedPage;  
import {connect} from 'react-redux';  
import {bindActionCreators} from 'redux';  
import { getUserInfo, verifyIdCard } from '../../../store/home/index';  
import {getManualAuditInfo} from '../../../store/user/index';  
import Component from './component';  
function mapStateToProps(state) {  
 return {  
 userInfo: state.userInfo  
 };  
}  
function matchDispatchToProps(dispatch) {  
 return bindActionCreators({getUserInfo, verifyIdCard, getManualAuditInfo}, dispatch);  
}  
export default connect(  
 mapStateToProps,  
 matchDispatchToProps,  
)(Component);  
import React, { useState, useEffect } from 'react';  
import { View, Text, StyleSheet, ScrollView, TouchableOpacity } from 'react-native';  
import Form from '../../Component/form';  
import ImageUpload from '../../Component/imageUpload';  
import { AppRoute } from '../../../navigator/AppRoutes';  
import { Spinner, Content } from 'native-base'  
import vertifyCn from '../config/IdCardVertifyCn';  
import Theme from '../../../style/colors';  
import HeaderCommon from '../../Component/HeaderCommon'  
export default function IdCardVertifyPage(props) {  
 const [formData, setFormData] = useState({});  
 const [formImages, setFormImages] = useState(['', '', '']);  
 const [userId, setUserId] = useState('');  
 const [loading, setLoading] = useState(false);  
 useEffect(() => {  
 setUId();  
 getAuthInfo();  
 }, []); //props.userInfo  
 function setUId() {  
 const { params } = props.route;  
 if (params && params.userId) {  
 setUserId(params.userId);  
 } else {  
 setUserId(props.userInfo.data.id)  
 }  
 }  
 function getAuthInfo() {  
 setLoading(true);  
 var uId = props.userInfo.data.id;  
 const { params } = props.route;  
 if (params && params.userId) {  
 uId = params.userId;  
 }  
 props.getManualAuditInfo(uId, res => {  
 console.log("1111111res1", res);  
 setLoading(false);  
 if (res.code === 0) {  
 if (!(res.data)  
 || (res.data.identificationType !== 'id\_card')) {  
 return;  
 }  
 let { name, identificationNo, gender, nation, birthDate, identificationAddress } = res.data;  
 var oldData = { name, identificationNo, gender, nation, birthDate, identificationAddress };  
 if (!oldData.gender) {  
 oldData.gender = 'male';  
 }  
 setFormData(oldData);  
 let imgs = [];  
 if (res.data.imageUrls && res.data.imageUrls.length) {  
 res.data.imageUrls.map((item, index) => {  
 $getImage(item, response => {  
 let data = Object.assign([], imgs);  
 data[index] = response;  
 setFormImages(data);  
 imgs[index] = response;  
 }, true)  
 })  
 }  
 } else {  
 console.log("222222222res2", res);  
 showToast(res.message);  
 }  
 });  
 }  
 const handleConfirm = () => {  
 let message = '';  
 console.log(111, formImages);  
 let index = vertifyCn.findIndex(item => {  
 return item.required && !formData[item.key];  
 });  
 message = vertifyCn[index] && vertifyCn[index].errorMsg[0];  
 console.log('\*\*\*\*3:', formData);  
 if (formData) {  
 var reg = /(^\d{15}$)|(^\d{18}$)|(^\d{17}(\d|X|x)$)/;  
 var nowDateTime = new Date();  
 var selDateTime = new Date(formData.birthDate);  
 if (!reg.test(formData.identificationNo)) {  
 message = '身份证输入不合法';  
 } else if (nowDateTime.getTime() < selDateTime.getTime()) {  
 message = '出生日期不能大于当前日期';  
 }  
 }  
 if (!message) {  
 console.log(111, formImages);  
 for (var i = 0; i < 3; i++) {  
 let item = formImages[i];  
 if (!item) {  
 message = '请上传三张图片';  
 break;  
 }  
 }  
 }  
 if (message) {  
 showToast(message);  
 return;  
 }  
 var result = new FormData();  
 changeToForm(result);  
 result.append('userId', userId);  
 result.append('identificationType', 'id\_card');  
 setLoading(true);  
 console.log('99userId: ', userId);  
 const controller = new AbortController(); //test  
 const { signal } = controller;  
 console.log('^^^^^^^^^', controller);  
 props.verifyIdCard(result, res => {  
 console.log('^^^^^^res1',res)  
 setLoading(false)  
 if (res.code === 0) {  
 showToast('提交成功');  
 NavigatorService.navigate(AppRoute.VERDETAILS);  
 const { params } = props.route;  
 if (params && params.refreshStatus) {  
 params.refreshStatus();  
 }  
 } else {  
 console.log('^^^^^^res2',res)  
 if (res === "timeout" || res.message === "Network request failed") {  
 } else {  
 showToast(res.message)  
 }  
 }  
 })  
 };  
 const changeToForm = (result) => {  
 for (var index in formData) {  
 result.append(index, formData[index]);  
 }  
 for (var i in formImages) {  
 result.append('images', formImages[i]);  
 }  
 };  
 const changeForm = data => {  
 setFormData(data);  
 };  
 const setImageForm = (index, obj) => {  
 let data = Object.assign([], formImages);  
 data[index] = obj;  
 setFormImages(data);  
 };  
 return (  
 loading ? <Spinner style={STYLES.spinner} color="#5C8BFF"/> :  
 <View style={styles.container}>  
 <HeaderCommon  
 options={{  
 backTitle: '返回',  
 title: '身份证认证'  
 }}  
 />  
 <Content>  
 <ScrollView bounces={false} contentContainerStyle={styles.scrollContainer}>  
 <Text style={styles.textTitle}>基本资料</Text>  
 <Form  
 config={vertifyCn}  
 class={styles.formBox}  
 changeForm={changeForm}  
 oldData={formData}  
 />  
 <Text style={styles.textTitle}>照片上传</Text>  
 <View style={styles.ImageUploadBox}>  
 <ImageUpload  
 title="身份证正面"  
 setImageForm={obj => setImageForm(0, obj)}  
 imgUrl={formImages[0] && formImages[0].uri}  
 />  
 <ImageUpload  
 title="身份证反面"  
 setImageForm={obj => setImageForm(1, obj)}  
 imgUrl={formImages[1] && formImages[1].uri}  
 />  
 <ImageUpload  
 title="手持身份证"  
 setImageForm={obj => setImageForm(2, obj)}  
 imgUrl={formImages[2] && formImages[2].uri}  
 />  
 </View>  
 <TouchableOpacity style={styles.Btn} onPress={() => { handleConfirm(); }}>  
 <Text style={styles.btnText}>确认</Text>  
 </TouchableOpacity>  
 </ScrollView>  
 </Content>  
 </View>  
 );  
}  
const styles = StyleSheet.create({  
 container: {  
 flex: 1,  
 },  
 scrollContainer: {  
 paddingTop: 16,  
 paddingHorizontal: 16,  
 backgroundColor: Theme.background,  
 },  
 textTitle: {  
 fontSize: $screen.scaleSize(16),  
 color: Theme.textDefault,  
 },  
 formBox: {  
 paddingBottom: 16,  
 marginTop: 10,  
 },  
 ImageUploadBox: {  
 flex: 1,  
 flexDirection: 'row',  
 justifyContent: 'space-between',  
 marginTop: 20,  
 },  
 Btn: {  
 marginTop: 100,  
 marginBottom: 50,  
 height: 40,  
 borderRadius: 20,  
 backgroundColor: Theme.primary,  
 },  
 btnText: {  
 height: 40,  
 lineHeight: 40,  
 textAlign: 'center',  
 fontSize: $screen.scaleSize(16),  
 color: '#FFFFFF',  
 },  
});  
import React, {useState} from 'react';  
import {  
 View,  
 Text,  
 Image  
} from 'react-native';  
export default function BlankPage(props) {  
 const img = require('../../assets/images/error.png')  
 const {errorMsg} = props;  
 return (  
 <View style={{flex: 1, alignItems: 'center', justifyContent: 'center'}}>  
 <Image source={img}></Image>  
 <Text style={{color: '#7C7C7C', fontSize: 14, marginTop: 16}}>{errorMsg}</Text>  
 </View>  
 )  
}