```
// DES427 Lab 08-09-10
import React from 'react';
import { Platform, StyleSheet, Text, View,
  TouchableHighlight, TextInput, Image, Alert, LogBox,
  ScrollView, Dimensions, ActivityIndicator, Button, TouchableOpacity } from
'react-native';
import { Picker } from '@react-native-picker/picker';
import Constants from 'expo-constants';
import * as Location from 'expo-location';

LogBox.ignoreLogs(['Setting a timer']);
LogBox.ignoreLogs(['AsyncStorage has been extracted from react-native core
and will be removed in a future release']); //
https://github.com/firebase/firebase-js-sdk/issues/1847

import { initializeApp } from 'firebase/app';
import { getDatabase, ref, onValue, get } from 'firebase/database';
import { getAuth, createUserWithEmailAndPassword, signInWithEmailAndPassword
} from "firebase/auth"; // https://github.com/firebase/firebase-js-
sdk/issues/1847

const firebaseConfig = {
  apiKey: "AIzaSyBd4UOMqos0OKJaZJ5U512XJijug1STCmk",
  authDomain: "des427-lab08.firebaseapp.com",
  databaseURL: "https://des427-lab08-default-rtdb.asia-
southeast1.firebasedatabase.app",
  projectId: "des427-lab08",
  storageBucket: "des427-lab08.appspot.com",
  messagingSenderId: "797708052260",
  appId: "1:797708052260:web:ead13f287dff7d9d3235b1",
  measurementId: "G-MJBZTTX9G3"
};

initializeApp(firebaseConfig);
var {width, height} = Dimensions.get('window');

// navigation stuffs
import { NavigationContainer } from '@react-navigation/native';
import { createStackNavigator } from '@react-navigation/stack';

const RADIUS = 20;

function Restaurant(props) {
  return (
    <TouchableHighlight onPress={()=>{
      props.navigation.navigate('Details',{ res_data: props.res_data,
changeState: props.changeState });
    }} underlayColor="white">
      <View style={styles.restaurant}>
        <Text style={{fontSize: 20}}>{props.res_data.name}</Text>
        <Text style={{fontSize: 12}}>วันทำการ {props.res_data.open_day} เวลา
ทำการ {props.res_data.open_time}</Text>
        <Text style={{fontSize: 12}}>โทรศัพท์ {props.res_data.phone}</Text>
        <View style={{flexDirection: 'row'}}>
          <Image source={{uri: props.res_data.images[0]}} style=
{styles.food_img}/>
          <Image source={{uri: props.res_data.images[1]}} style=
{styles.food_img}/>
```

```
          <Image source={{uri: props.res_data.images[2]}} style={styles.food_img}/>
        </View>
      </View>
    </TouchableHighlight>
  );
}

function Stars(props) {
  const star = parseInt(props.stars); let image = require(`./images/1-star.png`);
    switch(star){
      case 1: image = require(`./images/1-star.png`); break;
      case 2: image = require(`./images/2-star.png`); break;
      case 3: image = require(`./images/3-star.png`); break;
      case 4: image = require(`./images/4-star.png`); break;
      case 5: image = require(`./images/5-star.png`); break;
      default: image = require(`./images/1-star.png`); break;
    }
    return (<View style={styles.star_container}><Image source={image} style={styles.star_img}/></View>)
}

function StarWithAverage(props) {
  let stars_int = Math.round(props.stars);
  let stars2decimal = props.stars.toFixed(2);
  return (<View> <Text>Average star: {stars2decimal}</Text><Stars stars={stars_int}/> </View>)
}

function Reviews(props) {
  const reviews = props.reviews;
  return (
    <View style={{flex:1}}>
      <View style={{flexDirection:"row", justifyContent:"center"}}>
        <Text style={{fontSize:16}}>Reviews</Text></View>
      <ScrollView>{reviews.map((review,i) => {
          return (<View key={i}><Text>{review.comment}</Text><Stars stars={review.stars}/></View>)})}
        {/* workaround for scrollview cutoff at the bottom */}
      <Image source={require('./images/bottom_filler.png')}/>
      </ScrollView>
    </View>
  )
}

class SignupLogin extends React.Component {
  constructor(props){
    super(props);
    this.state = { username: '', password: '', confirmPassword: '', showLogin: true, };
  }

  toggleShowLogin() { this.setState({ showLogin: true }) }
  toggleShowSignup() { this.setState({ showLogin: false }) }

  doLogin() {
    const auth = getAuth();
    signInWithEmailAndPassword(auth, this.state.username, this.state.password).then( () => {
```

```
      console.log("login successful");
      this.props.loginCB();
    }) .catch(function(error) { alert(error.message); })
  }

  doSignup() {
    // https://firebase.google.com/docs/auth/web/password-auth
    const password = this.state.password;
    const confirmPassword = this.state.confirmPassword;
    if (password === confirmPassword){
      // do signup
      const auth = getAuth();
      createUserWithEmailAndPassword(auth, this.state.username,
this.state.password).then( () => {
        console.log("created new user successful"); this.toggleShowLogin();
// show login page
      }).catch(function(error) { alert(error.message); });
    }
    else { alert("Password do not match !!!"); }
  }

  showSignup() {
    return (
      <View>
        <View style={styles.group}>
          <Text style={styles.title}>Username</Text>
          <TextInput style={styles.input} value={this.state.username}
            onChangeText={(username) => this.setState({username})}/>
        </View>
        <View style={styles.group}>
          <Text style={styles.title}>Password</Text>
          <TextInput style={styles.input} secureTextEntry={true} value=
{this.state.password}
            onChangeText={(password) => this.setState({password})} />
        </View>
        <View style={styles.group}>
          <Text style={styles.title}>Confirm Password</Text>
          <TextInput style={styles.input} secureTextEntry={true} value=
{this.state.confirmPassword}
            onChangeText={(confirmPassword) =>
this.setState({confirmPassword})} />
        </View>
        <View style={styles.center}>
          <View style={styles.group}>
            <TouchableOpacity onPress={() => {this.toggleShowLogin();}}>
              <Text style={styles.signupText}>Login</Text>
            </TouchableOpacity>
          </View>
          <View style={styles.group}>
            <TouchableOpacity style={styles.button}
              onPress={() => {this.doSignup()}}>
              <Text style={styles.buttonText}>Signup</Text>
            </TouchableOpacity>
          </View>
        </View>
      </View>
    );
  }

  showLogin() {
```

```
    return (
      <View>
        <View style={styles.group}>
          <Text style={styles.title}>Username</Text>
          <TextInput style={styles.input} value={this.state.username}
onChangeText={(username) => this.setState({username})}/>
        </View>
        <View style={styles.group}>
          <Text style={styles.title}>Password</Text>
          <TextInput style={styles.input} secureTextEntry={true}
            value={this.state.password} onChangeText={(password) =>
this.setState({password})} />
        </View>
        <View style={styles.center}>
          <View style={styles.group}><TouchableOpacity onPress={() =>
{this.toggleShowSignup();}}>
            <Text style={styles.signupText}>Signup</Text> </TouchableOpacity>
          </View>
          <View style={styles.group}><TouchableOpacity style={styles.button}
onPress={() => {this.doLogin()}}>
            <Text style={styles.buttonText}>Login</Text></TouchableOpacity>
          </View>
        </View>
      </View>
    );
  }

  render() {
    return ( <View style={styles.containerLogin}> {this.state.showLogin ?
this.showLogin() : this.showSignup()} </View> );
  }
}

class HomeScreen extends React.Component {

  constructor(props) {
    super(props);
    this.restaurantRef = ref(getDatabase(), 'restaurantReview/');
    this.state = {
      searchText: '', isShowNearby: true, restaurantData: null, location:
null, errorMessage: null, searchResult: [],
      region: { latitude: 37.78825, longitude: -122.4324, latitudeDelta:
0.0922, longitudeDelta: 0.0421, },
      isLoggedIn: false, testState: { state1: 'brown', state2: 'fox' }
    };
    this.onRegionChangeComplete = this.onRegionChangeComplete.bind(this);
    this.loginSuccess = this.loginSuccess.bind(this);
    this._changeTestState = this._changeTestState.bind(this);
  }

  _changeTestState(newState) {
    console.log(newState);
    let resIdx = this.state.restaurantData.findIndex( (res) => { return
res.name === newState.resName; });
    console.log(this.state.restaurantData[resIdx].reviews);
    let newResData = this.state.restaurantData;
    newResData[resIdx].reviews.push({stars: newState.stars, comment:
newState.comment});
    this.setState({ restaurantData: newResData },()=>
{console.log(this.state.restaurantData)});
```

```
  }

  _onPressButton() {
    if(this.state.restaurantData) {
      // data is loaded from firebase
      const restaurantData = this.state.restaurantData; var restaurantFound = [];
      let patt = new RegExp(this.state.searchText, 'i'); // buid regex pattern
      // search for restaurant
      for (const element of restaurantData){
        let res_name = element.name; let res_type = element.type;
        let result = res_name.match(patt); let result2 = res_type.match(patt);
        if (result || result2) { restaurantFound.push(element) }
      }
      this.setState({ isShowNearby: false, searchResult: restaurantFound });
    }
  }

  onRegionChangeComplete(region) { this.setState({region}); }

  componentDidMount() {
    if (Platform.OS === 'android' && !Constants.isDevice) {
      this.setState({ errorMessage: 'Oops, this will not work on Sketch in an Android emulator. Try it on your device!', }); }
    else { this._getLocationAsync(); }
  }

  _getLocationAsync = async () => {
    let { status } = await Location.requestForegroundPermissionsAsync();
    if (status !== 'granted') { this.setState({ errorMessage: 'Permission to access location was denied', }); }
    let location = await Location.getCurrentPositionAsync({});
    // console.log(location);
    const region = { latitude: location.coords.latitude, longitude: location.coords.longitude,
      latitudeDelta: 0.0922, longitudeDelta: 0.0421, }
    this.setState({ region:region, location: location });
  };

  _readDB() {
    get(this.restaurantRef).then((snapshot) => {
      if (snapshot.exists()) { this.setState({restaurantData: snapshot.val()}) }
      else { console.log("No data available"); }
    }).catch((error) => { console.log(error); });
  }

  _computeDistance(lat1,lon1,lat2,lon2) {
    lat1 = parseFloat(lat1); lon1 = parseFloat(lon1);
    lat2 = parseFloat(lat2); lon2 = parseFloat(lon2);
    var toRadians = (deg) => { return deg * Math.PI/180; }
    // https://www.movable-type.co.uk/scripts/latlong.html
    var R = 6371e3; // metres
    var phi1 = toRadians(lat1); var phi2 = toRadians(lat2);
    var delta_phi = toRadians((lat2-lat1)); var delta_lambda = toRadians((lon2-lon1));
    var a = Math.sin(delta_phi/2) * Math.sin(delta_phi/2) + Math.cos(phi1) * Math.cos(phi2) *
```

```
      Math.sin(delta_lambda/2) * Math.sin(delta_lambda/2);
    var c = 2 * Math.atan2(Math.sqrt(a), Math.sqrt(1-a));
    var d = R * c;
    return d; // in metres
  }


  _isWithinRadius(latlong) {
    /*
    latlong is string of latlong, e.g. "13.6565217,100.6212236"
    this function computes the distance from current location
    (this.state.location.coords.latitude,
this.state.location.coords.longitude)
    to latlong and determine if it's within RADIUS km
    */
    var distance, lat, long;
    if (typeof(latlong) !== "string"){ throw new Error("latlong must be a
string"); }
    data = latlong.split(','); lat = parseFloat(data[0]); long =
parseFloat(data[1])

    // compute the distance from current location to latlong
    if (this.state.location) {
      currentLat = this.state.location.coords.latitude;
      currentLong = this.state.location.coords.longitude;
      distance = this._computeDistance(lat,long,currentLat,currentLong)
    }
    else {  throw new Error("location is not available yet"); }
    return distance <= RADIUS*1000;
  }

  showNearby() {
    if (this.state.restaurantData){
      const restaurantData = this.state.restaurantData;
      var nearbyRestaurantData = [];
      if (this.state.location != null) {
        restaurantData.forEach((element) => {
          if(this._isWithinRadius(element.gps)){
            nearbyRestaurantData.push(element);
        }});
      } return (
        <View style={styles.restaurantContainer}>
          <ScrollView style={{flex:1}}>
            {nearbyRestaurantData.map((res_data,i) => {
              return ( <Restaurant key={i} res_data={res_data}
                  navigation={this.props.navigation} changeState=
{this._changeTestState} /> ) })}
            <Image source={require('./images/bottom_filler.png')}/>
          </ScrollView>
        </View> )
    } else {
      return ( <View style={styles.restaurantContainer}>
          <Text>Please Wait</Text> <ActivityIndicator size="large"
color="#0000ff" />
        </View> )
    }
  }

  showSearchResult() {
    if (this.state.searchResult[0]){
```

```
      return (
        <View style={styles.restaurantContainer}>
          <ScrollView style={{flex:1}}>
            {this.state.searchResult.map((res_data,i) => {
              return ( <Restaurant key={i} res_data={res_data}
                  navigation={this.props.navigation} changeState=
{this._changeTestState} /> ) })}
            {/* workaround for scrollview cutoff at the bottom */}
            <Image source={require('./images/bottom_filler.png')}/>
          </ScrollView>
        </View>)
    } else { return (<Text>No restaurant found</Text>) }
  }

  showHome() {
    return (
      <View style={styles.container}>
        <Text style={styles.title}>Restaurant Search</Text>
        <View style={styles.searchArea}>
          <TextInput style={{height: 20, width:300, fontSize: 20}}
            placeholder="Search" onChangeText={(text) =>
this.setState({searchText: text})} />
          <TouchableHighlight onPress={()=>this._onPressButton()}
underlayColor="white">
            <View style={styles.searchButton}>
              <Image style={{height:30, width:30}} source=
{require('./images/search_icon.png')} />
            </View>
          </TouchableHighlight>
        </View>
        <View> {this.state.isShowNearby ? this.showNearby() :
this.showSearchResult()} </View>
      </View>
    )
  }

  showLogin() { return ( <SignupLogin loginCB={this.loginSuccess}/>) }
  loginSuccess() { this.setState({ isLoggedIn: true }, () => {
this._readDB(); }) }
  render() {
    return ( <View style={{flex:1}}> {this.state.isLoggedIn ? this.showHome()
: this.showLogin()} </View> );
  }
}

class DetailsScreen extends React.Component {

  constructor(props){
    super(props);
    /* 2. Read the params from the navigation state */
    const { res_data } = this.props.route.params;
    const { changeState } = this.props.route.params;
    const reviews = res_data.reviews;
    this.state = { showReviews: true, star: "1",
      res_data: res_data, reviews: reviews, changeState:changeState };
  }
  writeReview() { this.setState({ showReviews: false }) }
  showWriteReview(){
    return(
      <View style={{flex: 1, margin:10, padding:10}}>
```

```
        <View style={{height:20}}/>
        <Text>Stars</Text>
        <Picker selectedValue={this.state.star} style={{height: 50, width:
200}}
          onValueChange={(itemValue) => this.setState({star: itemValue}) }>
          <Picker.Item label="1 star" value="1" />
          <Picker.Item label="2 star" value="2" />
          <Picker.Item label="3 star" value="3" />
          <Picker.Item label="4 star" value="4" />
          <Picker.Item label="5 star" value="5" />
        </Picker>
        <TextInput multiline={true} numberOfLine={4}
          style={{height:100, borderColor: 'gray', borderWidth: 1}}
          value={this.state.comment} onChangeText={(comment) =>
this.setState({comment})} />
        <View style={styles.group}>
          <TouchableOpacity style={styles.button} onPress={() =>
{this._onPressSubmitReview();}}>
            <Text style={styles.buttonText}>Review</Text>
          </TouchableOpacity>
        </View>
      </View>
    )
  }

  _onPressSubmitReview() {
    // show stars and comments in alert box
    Alert.alert(this.state.star, this.state.comment);
    this.state.changeState({ stars: this.state.star, comment:
this.state.comment,
      resName : this.state.res_data.name })
    this.setState({ showReviews: true })
  }

  showReviews(res_data, average_star, reviews) {
    return (
      <View style={[{ flex: 1, alignItems: 'center', justifyContent:
'center', }, styles.restaurantContainer]}>
        <Restaurant key={1} res_data={res_data} navigation=
{this.props.navigation} changeState={null} />
        {/* show star picture and average star value */}
        <StarWithAverage stars={average_star}/>
        {/* show review texts */}
        <TouchableOpacity onPress={() => {this.writeReview();}}>
          <Text style={styles.signupText}>Write Review</Text>
        </TouchableOpacity>
        <Reviews reviews={reviews}/>
      </View>
    )
  }

  render() {
    let sum_stars = 0;
    for (const review of this.state.reviews) { sum_stars +=
parseInt(review.stars); }
    let average_star = sum_stars/this.state.reviews.length;
    return (
      <View style={{flex:1}}>
        {this.state.showReviews ? this.showReviews(this.state.res_data,
average_star, this.state.reviews) : this.showWriteReview()}
```

```
      </View>
    );
  }
}

const Stack = createStackNavigator();

export default class App extends React.Component {
  render() {
    return (
      <NavigationContainer>
        <Stack.Navigator initialRouteName="Home">
          <Stack.Screen name="Home" component={HomeScreen}
            options={{ title: 'Restaurants',
              headerStyle: { backgroundColor: '#f4511e', },
              headerTintColor: '#fff', headerTitleStyle: { fontWeight:
'bold', }, }} />
          <Stack.Screen name="Details" component={DetailsScreen} />
        </Stack.Navigator>
      </NavigationContainer>
    );
  }
}

const styles = StyleSheet.create({
  container: { flex: 1, backgroundColor: '#fff', alignItems: 'center',
justifyContent: 'flex-start', padding: 20 },
  title: { fontSize: 20, padding: 10 },
  searchArea: { flexDirection: "row", padding: 10, backgroundColor:
'#E5E4E3', borderRadius: 10, alignItems: 'center' },
  restaurantContainer: { padding: 5, margin: 10, backgroundColor: '#E5E4E3',
width: 350, flex:1 },
  restaurant: { padding: 5, margin: 5, backgroundColor: '#FFFFFF', },
  food_img: { width: 100, height: 100, margin: 3 },
  star_img : { width: 120, height:30, margin: 3 },
  star_container : { padding: 5, margin: 5, flexDirection : "row",
backgroundColor: '#FFFFFF', alignItems: "center" },
  containerLogin: { flex: 1, justifyContent: 'center', flexDirection:
'column', padding: 20 },
  group: { marginTop: 20 },
  button: { backgroundColor: 'lightblue', padding: 20, borderWidth: 1 },
  buttonText: { fontSize: 30 },
  input: { padding: 10, height: 40, borderWidth: 1 },
  title: { fontSize: 20 },
  center: { alignItems: 'center' },
  signupText : { fontSize: 20, color: 'blue' }
});

```

```
// Lab 11 Functional Components

import React, { useState, useEffect } from 'react';
import { StyleSheet, Text, View, Dimensions } from 'react-native';
import { LineChart } from 'react-native-chart-kit';
import StockButton from './StockButton';
import API from './api';
import Switch from './switch';

const chartConfig = {
  backgroundGradientFrom: '#1E2923',
  backgroundGradientTo: '#08130D',
  color: (opacity = 1) => `rgba(26, 255, 146, ${opacity})`, // color of background
  strokeWidth: 2 // optional, default 3
}

export default function Stocks(props){
  const [loading, setLoading] = useState(false);
  const [stockCode, setStockCode] = useState('Not Chosen Yet');
  const [stockName, setStockName] = useState('N/A');
  const [stockMode, setStockMode] = useState('Daily');
  const [dates, setDates] = useState(["01/01", "02/01", "03/01", "04/01", "05/01", "06/01", "07/01"]);
  const [prices, setPrices] = useState([1,2,3,4,5,6,7]);
  const [switchValue, setSwitchValue] = useState(false);

  useEffect(() => {
    console.log("Calling state: ", stockCode, stockMode);
    // Loading data from API
    if (stockName =/= 'N/A') {
      setLoading(true);
      API(stockCode,stockMode).then((stock) => {
        if(stock["Time Series (Daily)"] =/= undefined || stock["Weekly Time Series"] =/= undefined){
          let datesArray = stockMode === "Daily" ? Object.keys(stock["Time Series (Daily)"]).slice(0,7) : Object.keys(stock["Weekly Time Series"]).slice(0,7);
          let closingPrice = [];
          datesArray.forEach((day) => {
            closingPrice.push(stockMode === "Daily" ? stock["Time Series (Daily)"][day]["4. close"] : stock["Weekly Time Series"][day]["4. close"]);
          });
          let datesArrayRev = datesArray.reverse();
          let dayMonthArray = [];
          datesArrayRev.forEach((element) => { let dayArray = element.split('-'); dayMonthArray.push(dayArray[2]+'/'+dayArray[1]); });
          setDates(dayMonthArray); setPrices(closingPrice);
        }else{ alert("Data is not return due to exceeding the API call limit. Please try again later."); }
      }).catch(function(error){ console.log(error); }).finally(() => setLoading(false)); }
  }, [stockCode, stockMode]);

  const changeIndex = (stockCode,stockName) =>{ setStockCode(stockCode); setStockName(stockName); }
  const handleSwitchChange = async () => {
    let newSwitchValue = !switchValue;
```

```
    setSwitchValue(newSwitchValue); setStockMode(newSwitchValue ? "Weekly" : "Daily");
    changeIndex(stockCode,stockName);
  }

  return(
      <View style={styles.container}>
        <View style={styles.header}>
          <Text style={styles.title}>Company: {stockCode} / Mode: {stockMode}</Text>
          <LineChart
            data={{ labels: dates,
              datasets: [{ data: prices,
                color: (opacity = 1) => `rgba(134, 65, 244, ${opacity})`, strokeWidth: 2 }]
            }} width={Dimensions.get('window').width}
            height={400} chartConfig={chartConfig} style={{paddingVertical:10}} />
        </View>
        <View style={[styles.rowContainer,styles.footer]}>
          <View style={[styles.rowContainer]}>
            <Switch onValueChange={handleSwitchChange} value={switchValue}></Switch>
          </View>
          <View style={styles.rowContainer}>
            <StockButton code="AAPL" name="Apple" onPress={changeIndex} disabled={loading}></StockButton>
            <StockButton code="GOOGL" name="Google" onPress={changeIndex} disabled={loading}></StockButton>
            <StockButton code="UBER" name="Uber" onPress={changeIndex} disabled={loading}></StockButton>
          </View>
          <Text style={styles.textWhite}>{loading ? "New Data is loading..." : stockName === "N/A" ? "No Company is chosen yet" : stockCode + " loaded successfully."}</Text>
        </View>
      </View>
    );
}

const styles = StyleSheet.create({
  container: { flex: 1, paddingTop: 40},
  title:{ flex: 2, marginTop: 15, fontSize: 24, color: 'white', fontWeight: 'bold'},
  header:{ flex: 2, justifyContent: 'center', alignItems: 'center', backgroundColor: 'black'},
  rowContainer:{ flexDirection: 'row', justifyContent:'space-between' },
  columnContainer:{ flexDirection: 'column', justifyContent: 'space-between' },
  footer:{ flex: 1, flexDirection: 'row', flexWrap: 'wrap', alignItems:'center',
    justifyContent: 'space-evenly', backgroundColor: 'gray' },
  button:{ margin: 10, borderWidth: 1, width: 100,
    height: 50, borderRadius: 10, justifyContent: 'center',
    alignItems: 'center', backgroundColor: 'lightgray'},
  textWhite:{ color: 'white' }
});

```

```
// Lab 12: Recoil App.js
import * as React from "react";
import { RecoilRoot } from "recoil";
import { AppList } from "./AppList";

export default function App() {
  return ( <RecoilRoot> <AppList /> </RecoilRoot> );
}

// Lab 12: Recoil AppList.js
import React from "react";
import { SafeAreaView, View, StyleSheet, StatusBar } from "react-native";
import { BlueListPanel, RedListPanel } from "./ListPanel";

const styles = StyleSheet.create({
    container: { flex: 1, flexDirection: 'column', justifyContent: "space-
between" },
    row:{ flex: 1, paddingTop: 20, flexDirection: 'row', justifyContent:
"space-between"}
});

export function AppList() {
    return (
    <SafeAreaView style={styles.container}>
        <StatusBar/>
        <View style={styles.row}>
            <RedListPanel title={"Red List"} subtitle={"Top Left"} />
            <BlueListPanel title={"Blue List"} subtitle={"Top Right"} />
        </View>
        <View style={styles.row}>
            <BlueListPanel title={"Blue List"} subtitle={"Bottom Left"} />
            <RedListPanel title={"Red List"} subtitle={"Bottom Right"} />
        </View>
    </SafeAreaView>
    );
}

// Lab 12 Recoil RecoilListState.js
import {atom} from 'recoil';
export const redListState = atom({ key: 'redListState', default: [] });
export const blueListState = atom({ key: 'blueListState', default: []});

```

```
// Lab 12 Recoil ListItem.js
import * as React from "react";
import { useRecoilState } from "recoil";
import { StyleSheet, TouchableOpacity, Text, View } from "react-native";
import { redListState, blueListState } from "./RecoilListState";

const styles = StyleSheet.create({
  container: {flex: 1, flexDirection: 'row'},
  input: { paddingHorizontal: 8, marginRight: 4, backgroundColor: "white", height: 40, flex: 1, color:'#333', },
  deleteRedButton: {backgroundColor: 'pink', padding: 5, alignItems: 'center', borderRadius: 5, justifyContent: 'center', flex: 1},
  deleteBlueButton: {backgroundColor: 'lightblue', padding: 5, alignItems: 'center', borderRadius: 5, justifyContent: 'center', flex: 1},
  deleteButtonLabel: { color: "black", fontWeight: "bold"},
  listText: {paddingVertical: 10, flex: 9}
});

export function RedListItem({item}){
    const [redList, setRedList] = useRecoilState(redListState);
    const itemIndex = redList.findIndex((listItem) => listItem.id === item.id);

    const handleDelete = () => {
        console.log("Red Delete Pressed Index: ", itemIndex);
        setRedList(redList.filter((item, idx) => idx !== itemIndex));
    };

    return(
    <View style={styles.container}>
      <Text style={styles.listText}>#{itemIndex+1} | {item.text}</Text>
      <TouchableOpacity style={styles.deleteRedButton} onPress={handleDelete}>
          <Text style={styles.deleteButtonLabel}>x</Text>
      </TouchableOpacity>
    </View>
    );
}

export function BlueListItem({item}){
    const [blueList, setBlueList] = useRecoilState(blueListState);
    const itemIndex = blueList.findIndex((listItem) => listItem.id === item.id);

    const handleDelete = () => {
        console.log("Blue Delete Pressed Index: ", itemIndex);
        setBlueList(blueList.filter((item, idx) => idx !== itemIndex));
    };

    return(
    <View style={styles.container}>
      <Text style={styles.listText}>#{itemIndex+1} | {item.text}</Text>
      <TouchableOpacity style={styles.deleteBlueButton} onPress={handleDelete}>
          <Text style={styles.deleteButtonLabel}>x</Text>
      </TouchableOpacity>
    </View>
    );
}
```

```jsx
import React, { useState } from "react";
import { FlatList, View, StyleSheet, Text, TextInput, TouchableOpacity } from "react-native";
import { useRecoilState } from "recoil";
import { redListState, blueListState } from "./RecoilListState";
import { RedListItem, BlueListItem } from "./ListItem";

const styles = StyleSheet.create({
    title: {fontSize: 18, fontWeight: 'bold'},
    subtitle: {fontSize: 16},
    box: { flex: 1, marginHorizontal: 20, alignItems: 'flex-start', justifyContent: 'center'},
    flatListContainer: {flex: 1, flexDirection: 'row'},
    flatList: {marginVertical: 10},
    inputContainer: {paddingTop: 10, flexDirection: 'row', justifyContent: 'flex-start'},
    input: { flex: 3, height: 40, borderWidth: 1, borderTopLeftRadius: 5, borderBottomLeftRadius: 5, padding: 10 },
    button: { flex: 1, height: 40, alignItems: "center", padding: 10, borderTopRightRadius: 5, borderBottomRightRadius: 5 },
    borderColorBlue: { borderColor: 'blue' },
    borderColorRed: { borderColor: 'red' },
    backgroundColorBlue: { backgroundColor: 'blue' },
    backgroundColorRed: { backgroundColor: 'red' },
    textWhite:{ color: 'white' }
});

export function RedListPanel(props){

    const [redInput, setRedInput] = useState("");
    const [redList, setRedList] = useRecoilState(redListState);

    const handleRedButtonPress = () => {
        if (redInput === "") return;
        console.log("Red Button Pressed, New Item: ", redInput);
        setRedList([...redList, {id: redList.length + 1, text: redInput}]);
        setRedInput("");
    }

    return (
        <View style={styles.box}>
            <Text style={{color: 'red', fontSize: 18, fontWeight: 'bold'}}>{props.title}</Text>
            <Text style={styles.subtitle}>{props.subtitle}</Text>
            <View style={styles.inputContainer}>
                <TextInput style={[styles.input, styles.borderColorRed]} value={redInput} onChangeText={setRedInput} placeholder="Add a red item" />
                <TouchableOpacity style={[styles.button, styles.backgroundColorRed]} onPress={handleRedButtonPress}><Text style={styles.textWhite}>Add</Text></TouchableOpacity>
            </View>
            <View style={styles.flatListContainer}>
                <FlatList
                    data={redList}
                    style={styles.flatList}
                    renderItem={({item}) => <RedListItem item={item} />}
                    keyExtractor={(item) => `${item.id}`}
                />
            </View>
```

```
        </View>
    );
}

export function BlueListPanel(props){

    const [blueInput, setBlueInput] = useState("");
    const [blueList, setBlueList] = useRecoilState(blueListState);

    const handleBlueButtonPress = () => {
        if (blueInput === "") return;
        console.log("Blue Button Pressed, New Item: ", blueInput);
        setBlueList([...blueList, {id: blueList.length + 1, text:
blueInput}]);
        setBlueInput("");
    }

    return (
        <View style={styles.box}>
            <Text style={{color: 'blue', fontSize: 18, fontWeight: 'bold'}}>
{props.title}</Text>
            <Text style={styles.subtitle}>{props.subtitle}</Text>
            <View style={styles.inputContainer}>
                <TextInput style={[styles.input, styles.borderColorBlue]}
value={blueInput} onChangeText={setBlueInput} placeholder="Add a blue item" />
                <TouchableOpacity style={[styles.button,
styles.backgroundColorBlue]} onPress={handleBlueButtonPress}><Text style=
{styles.textWhite}>Add</Text></TouchableOpacity>
            </View>
            <View style={styles.flatListContainer}>
                <FlatList
                    style={styles.flatList}
                    data={blueList}
                    renderItem={({item}) => <BlueListItem item={item} />}
                    keyExtractor={(item) => `${item.id}`}
                />
            </View>
        </View>
    );
}
```