



# **COMSATS UNIVERSITY ISLAMABAD, ATTOCK CAMPUS**

**DEPARTMENT OF COMPUTER SCIENCES**

## **ASSIGNMENT NO: 4**

**SUBMITTED BY:**

**EMAN CHAUDHARY**

**REGISTRATION NO:**

**SP22-BSE-031**

**SUBMITTED TO:**

**Mr. MUHAMMAD KAMRAN**

**SUBJECT:**

**MOBILE APP DEVELOP**

**DATE:**

**24 DECEMBER, 2024**

## QUESTION:

- Replace the `useState` in assign 3 with `redux`. Store all states in global store and create necessary actions and reducers to update the state.
- Store the user data in `Firebase` when user sign up. And add necessary checks when the user login.
- Fetch the user location when they login and store that location in the `AsyncStorage`. Also display it in a component

## CODE:

```
import React, { useEffect } from 'react';
import { View, Text, StyleSheet, TextInput, TouchableOpacity, Image, ScrollView, FlatList, Alert } from 'react-native';
import { NavigationContainer } from '@react-navigation/native';
import { createStackNavigator } from '@react-navigation/stack';
import { Provider, useDispatch, useSelector } from 'react-redux';
import { createStore } from 'redux';
import AsyncStorage from '@react-native-async-storage/async-storage';
import * as Location from 'expo-location';

// Firebase API Endpoint

const FIREBASE_API_ENDPOINT = 'https://first-project-a348a-default-rtdb.firebaseio.com/';

// Sample data for recent orders

const recentOrders = [
  {
    id: '1',
    itemName: 'Beef Burger, French Fries, Cheese Burst',
    price: '$12.99',
    imageUrl: 'https://upload.wikimedia.org/wikipedia/commons/0/0b/RedDot_Burger.jpg',
  },
  {
    id: '2',
    itemName: 'Chicken Burger, French Fries, Pepsi',
    price: '$10.99',
    imageUrl: 'https://upload.wikimedia.org/wikipedia/commons/0/0b/RedDot_Burger.jpg',
  },
  {
    id: '3',
    itemName: 'Cheese Pizza, Garlic Bread',
    price: '$14.99',
    imageUrl: 'https://upload.wikimedia.org/wikipedia/commons/d/d1/Pepperoni_pizza.jpg',
  },
];

// Actions
```

```

const SET_USER = 'SET_USER';
const SET_LOCATION = 'SET_LOCATION';

const setUser = (userData) => ({
  type: SET_USER,
  payload: userData,
});

const setLocation = (location) => ({
  type: SET_LOCATION,
  payload: location,
});

// Reducers
const initialState = {
  user: {
    username: '',
    email: '',
    phone: '',
    password: '',
  },
  location: null,
};

const rootReducer = (state = initialState, action) => {
  switch (action.type) {
    case SET_USER:
      return { ...state, user: action.payload };
    case SET_LOCATION:
      return { ...state, location: action.payload };
    default:
      return state;
  }
};

// Store
const store = createStore(rootReducer);

// Profile Screen Component
function ProfileScreen() {
  const location = useSelector((state) => state.location);

  return (
    <ScrollView style={styles.container}>
      <View style={styles.profileContainer}>
        <View style={styles.profileDetails}>
          <Image
            source={{
              uri:
                'https://upload.wikimedia.org/wikipedia/commons/7/7c/Profile_avatar_placeholder_large.png'
            }}
            style={styles.profileImage}
          />
        </View>
      </View>
    </ScrollView>
  );
}

```

```

    <View>
      <Text style={styles.name}>EMAN CH</Text>
      <View style={styles.statsContainer}>
        <View style={styles.statsBox}>
          <Text style={styles.statsText}>1,410</Text>
          <Text style={styles.statsLabel}>Points</Text>
        </View>
        <View style={styles.statsBox}>
          <Text style={styles.statsText}>30+</Text>
          <Text style={styles.statsLabel}>Orders</Text>
        </View>
      </View>
    </View>
  </View>
</View>

<View style={styles.sectionHeader}>
  <Text style={styles.sectionTitle}>Recent Orders</Text>
  <Text style={styles.viewAll}>See All</Text>
</View>
<FlatList
  data={recentOrders}
  keyExtractor={({item}) => item.id}
  renderItem={({ item }) => (
    <View style={styles.orderItem}>
      <Image source={{ uri: item.imageUrl }} style={styles.orderImage} />
      <View style={styles.orderDetails}>
        <Text style={styles.itemName}>{item.itemName}</Text>
        <Text style={styles.itemPrice}>{item.price}</Text>
      </View>
    </View>
  )}
/>
{location && (
  <View style={styles.locationContainer}>
    <Text style={styles.locationText}>Location: {location.latitude},
{location.longitude}</Text>
  </View>
)}
</ScrollView>
);
}

```

*// Signup Screen Component*

```

function SignupScreen({ navigation }) {
  const dispatch = useDispatch();
  const [username, setUsername] = React.useState('');
  const [email, setEmail] = React.useState('');
  const [phone, setPhone] = React.useState('');
  const [password, setPassword] = React.useState('');

  // Mock registered emails for demonstration
  const registeredEmails = ['example@gmail.com', 'user@yahoo.com'];

```

```

const validateSignup = () => {
  const emailRegex = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;
  const phoneRegex = /^+92-3\d{2}-\d{7}$/;
  const usernameRegex = /^[a-zA-Z]+$/;
  const passwordRegex = /^(?=.*[A-Z])(?=.*\d)(?=.*[@$!%*?&])[A-Za-z\d@$!%*?&]{6,}$/;

  if (!usernameRegex.test(username)) {
    Alert.alert('Invalid Username', 'Username must contain alphabets only.');
```

return false;

```

  }
  if (username.length < 3 || username.length > 20) {
    Alert.alert(
      'Invalid Username',
      'Username must be between 3 and 20 characters long.'
    );
    return false;
  }
  if (!emailRegex.test(email)) {
    Alert.alert('Invalid Email', 'Please enter a valid email address.');
```

return false;

```

  }
  if (registeredEmails.includes(email)) {
    Alert.alert(
      'Email Already Registered',
      'This email address is already in use. Please log in or use a different email.'
    );
    return false;
  }
  if (!phoneRegex.test(phone)) {
    Alert.alert('Invalid Phone Number', 'Phone number must match +92-3xx-xxxxxxx.');
```

return false;

```

  }
  if (!passwordRegex.test(password)) {
    Alert.alert(
      'Weak Password',
      'Password must be at least 6 characters long and include at least one uppercase
letter, one number, and one special character.'
    );
    return false;
  }
  return true;
};

const handleSignup = async () => {
  if (validateSignup()) {
    const userData = { username, email, phone, password };

    // Save user data to Firebase
    try {
      const response = await fetch(`${FIREBASE_API_ENDPOINT}/users.json`, {
        method: 'POST',
        headers: {
          'Content-Type': 'application/json',
        },
      });
    }
  }
};

```

```

        body: JSON.stringify(userData),
    });
    const result = await response.json();
    if (result) {
        dispatch(setUser(userData));
        Alert.alert('Signup Successful', 'You can now log in.');
```

navigation.navigate('Login');

```

    } else {
        Alert.alert('Error', 'There was an issue with the signup. Please try again.');
```

}

```

    } catch (error) {
        Alert.alert('Error', 'Failed to save data. Please try again later.');
```

}

```

    }
};

return (
    <ScrollView contentContainerStyle={styles.formContainer}>
        <Text style={styles.formTitle}>Signup</Text>
        <TextInput
            style={styles.input}
            placeholder="Username"
            onChangeText={setUsername}
            value={username}
        />
        <TextInput
            style={styles.input}
            placeholder="Email"
            keyboardType="email-address"
            onChangeText={setEmail}
            value={email}
        />
        <TextInput
            style={styles.input}
            placeholder="Phone (+92-3xx-xxxxxxx)"
            keyboardType="phone-pad"
            onChangeText={setPhone}
            value={phone}
        />
        <TextInput
            style={styles.input}
            placeholder="Password"
            secureTextEntry
            onChangeText={setPassword}
            value={password}
        />
        <TouchableOpacity style={styles.button} onPress={handleSignup}>
            <Text style={styles.buttonText}>Signup</Text>
        </TouchableOpacity>
    </ScrollView>
);
}

// Login Screen Component

```

```

function LoginScreen({ navigation }) {
  const dispatch = useDispatch();
  const [email, setEmail] = React.useState('');
  const [password, setPassword] = React.useState('');

  const validateLogin = () => {
    if (email.trim() === '' || password.trim() === '') {
      Alert.alert('Invalid Input', 'Both fields are required.');
```

return false;

```
    }
    return true;
  };

  const handleLogin = async () => {
    if (validateLogin()) {
      // Fetch and store Location
      const { status } = await Location.requestForegroundPermissionsAsync();
      if (status === 'granted') {
        const location = await Location.getCurrentPositionAsync({});
        await AsyncStorage.setItem('location', JSON.stringify(location.coords));
        dispatch(setLocation(location.coords));
      }
      navigation.navigate('Profile');
```

};

```

  return (
    <ScrollView contentContainerStyle={styles.formContainer}>
      <Text style={styles.formTitle}>Login</Text>
      <TextInput
        style={styles.input}
        placeholder="Email"
        keyboardType="email-address"
        onChangeText={setEmail}
        value={email}
      />
      <TextInput
        style={styles.input}
        placeholder="Password"
        secureTextEntry
        onChangeText={setPassword}
        value={password}
      />
      <TouchableOpacity style={styles.button} onPress={handleLogin}>
        <Text style={styles.buttonText}>Login</Text>
      </TouchableOpacity>
    </ScrollView>
  );
}

// React Navigation Setup

const Stack = createStackNavigator();

```

```
function App() {
  return (
    <NavigationContainer>
      <Stack.Navigator initialRouteName="Signup">
        <Stack.Screen name="Signup" component={SignupScreen} />
        <Stack.Screen name="Login" component={LoginScreen} />
        <Stack.Screen name="Profile" component={ProfileScreen} />
      </Stack.Navigator>
    </NavigationContainer>
  );
}
```

```
export default function MainApp() {
  return (
    <Provider store={store}>
      <App />
    </Provider>
  );
}
```

// Styles

```
const styles = StyleSheet.create({
  container: {
    flex: 1,
    padding: 20,
    backgroundColor: '#E9F5F2',
  },
  profileContainer: {
    backgroundColor: 'ffffff',
    borderRadius: 20,
    padding: 20,
    marginBottom: 30,
    shadowColor: '#000',
    shadowOpacity: 0.15,
    shadowOffset: { width: 0, height: 4 },
    shadowRadius: 12,
    elevation: 8,
  },
  profileDetails: {
    flexDirection: 'row',
    alignItems: 'center',
  },
  profileImage: {
    width: 80,
    height: 80,
    borderRadius: 40,
    marginRight: 20,
    borderColor: '#f4a261',
    borderWidth: 2,
  },
  name: {
    fontSize: 22,
```



```
fontWeight: 'bold',
color: '#264653',
},
statsContainer: {
  flexDirection: 'row',
  marginTop: 10,
},
statsBox: {
  marginRight: 25,
  alignItems: 'center',
},
statsText: {
  fontSize: 18,
  fontWeight: '600',
  color: '#2a9d8f',
},
statsLabel: {
  fontSize: 13,
  color: '#6c757d',
},
sectionHeader: {
  flexDirection: 'row',
  justifyContent: 'space-between',
  alignItems: 'center',
  marginBottom: 20,
},
sectionTitle: {
  fontSize: 18,
  fontWeight: 'bold',
  color: '#264653',
},
viewAll: {
  fontSize: 14,
  color: '#e76f51',
  fontWeight: '600',
},
orderItem: {
  flexDirection: 'row',
  backgroundColor: '#fff',
  borderRadius: 15,
  padding: 15,
  marginBottom: 15,
  shadowColor: '#000',
  shadowOpacity: 0.1,
  shadowOffset: { width: 0, height: 2 },
  shadowRadius: 10,
  elevation: 4,
},
orderImage: {
  width: 60,
  height: 60,
  borderRadius: 10,
  marginRight: 15,
},
```

```
orderDetails: {
  flex: 1,
  justifyContent: 'center',
},
itemName: {
  fontSize: 16,
  fontWeight: '500',
  color: '#264653',
},
itemPrice: {
  fontSize: 14,
  fontWeight: '400',
  color: '#2a9d8f',
  marginTop: 5,
},
formContainer: {
  flexGrow: 1,
  justifyContent: 'center',
  padding: 20,
  backgroundColor: '#E9F5F2',
},
formTitle: {
  fontSize: 24,
  fontWeight: 'bold',
  marginBottom: 20,
  color: '#264653',
  textAlign: 'center',
},
input: {
  height: 50,
  borderColor: '#2a9d8f',
  borderWidth: 1,
  borderRadius: 10,
  marginBottom: 20,
  paddingHorizontal: 15,
  fontSize: 16,
},
button: {
  backgroundColor: '#2a9d8f',
  paddingVertical: 15,
  borderRadius: 10,
  alignItems: 'center',
},
buttonText: {
  color: '#fff',
  fontSize: 16,
  fontWeight: '600',
},
locationContainer: {
  marginTop: 20,
  padding: 10,
  backgroundColor: '#f4f4f4',
  borderRadius: 8,
  borderWidth: 1,
```

```

borderColor: '#ccc',
shadowColor: '#000',
shadowOffset: { width: 0, height: 2 },
shadowOpacity: 0.1,
shadowRadius: 4,
elevation: 2,
alignItems: 'center',
justifyContent: 'center',
},
locationText: {
  fontSize: 14,
  fontWeight: 'bold',
  color: '#333',
  textAlign: 'center',
  textTransform: 'uppercase',
  letterSpacing: 1,
},
});

```

## SCREENS:

Signup

EmanChaudhary

Eman@gmail.com

+92-343-5961184

\*\*\*\*\*

Signup

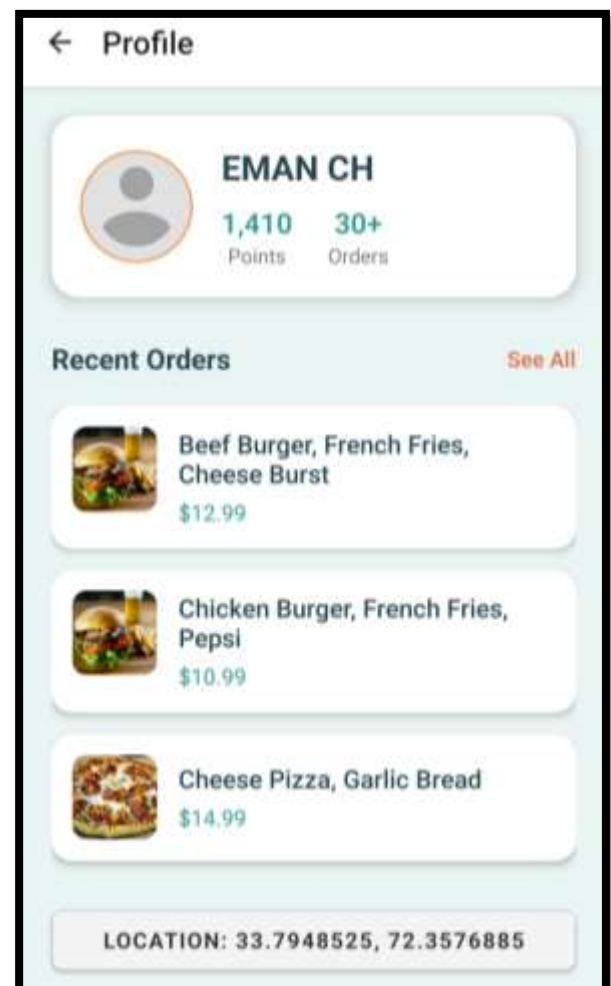
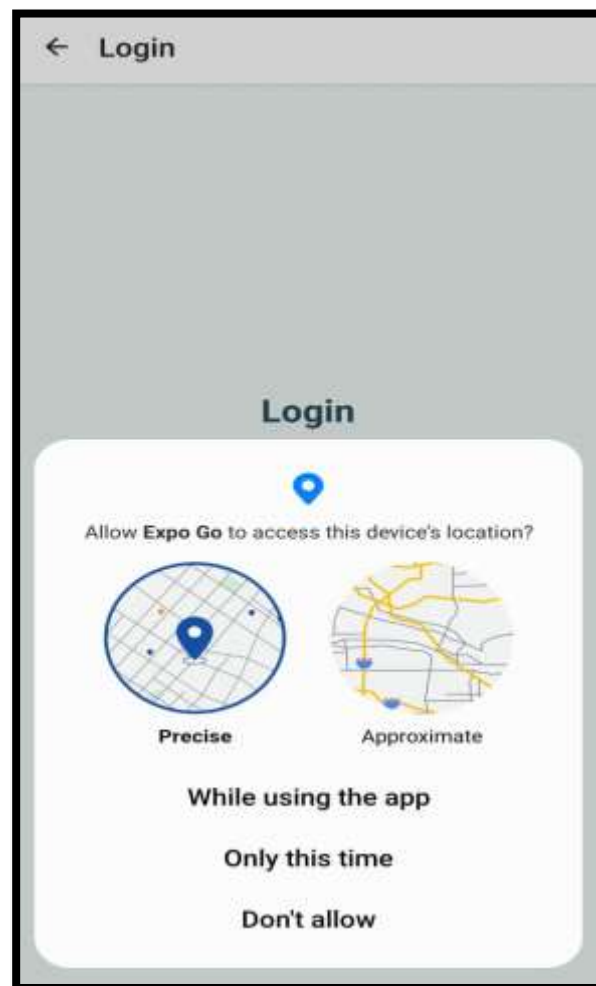
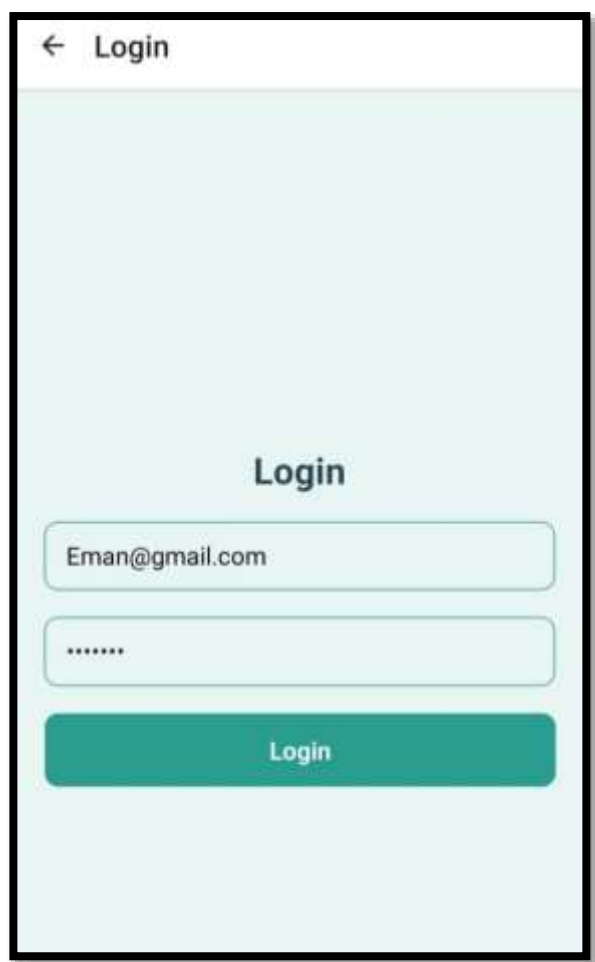
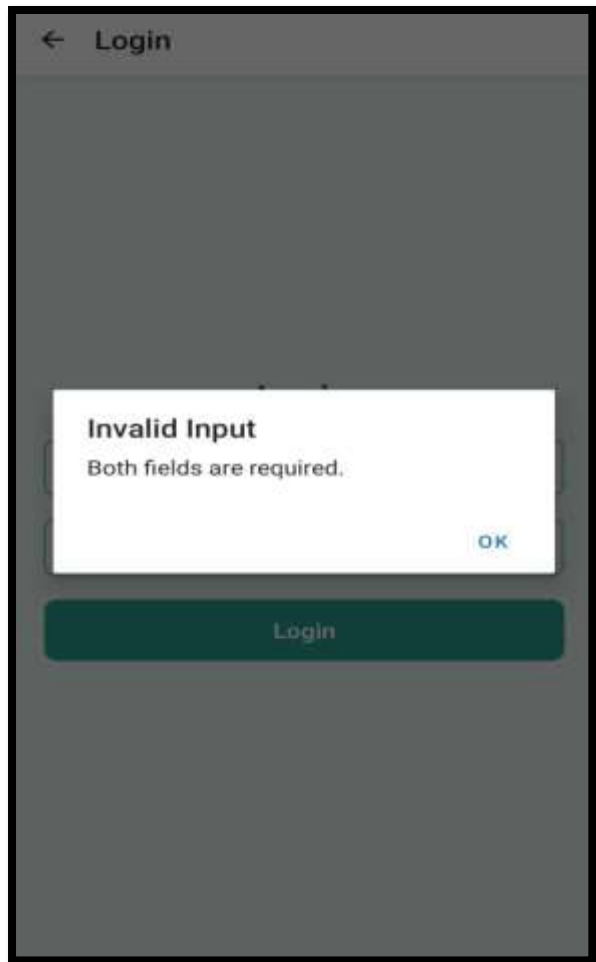
Login

Signup Successful

You can now log in.


OK

Login



# DATA IN FIREBASE:

 <https://first-project-a348a-default-rtdb.firebaseio.com>

 **Your security rules are defined as public, so anyone can steal, modify or delete data in your database**

<https://first-project-a348a-default-rtdb.firebaseio.com/>

▼ — users

- ▶ — -OEAlF-dGL12dUkxW0dy
- ▶ — -OEAlF7rtoDyLPIwtsJ8
- ▶ — -OEAlJLYdh60DVkpxzXd
- ▶ — -OETemxxsJBGh6hxYv15
- ▶ — -OETgB-gADyvejmoZWfu

▼ — -OEU2X-hJHymyzq5tQFG

- email: "Eman@gmail.com"
- password: "Aa@1234"
- phone: "+92-343-5961184"
- username: "EmanChaudhary"