This document is a lab report that documents reflection material

1 Screenshots

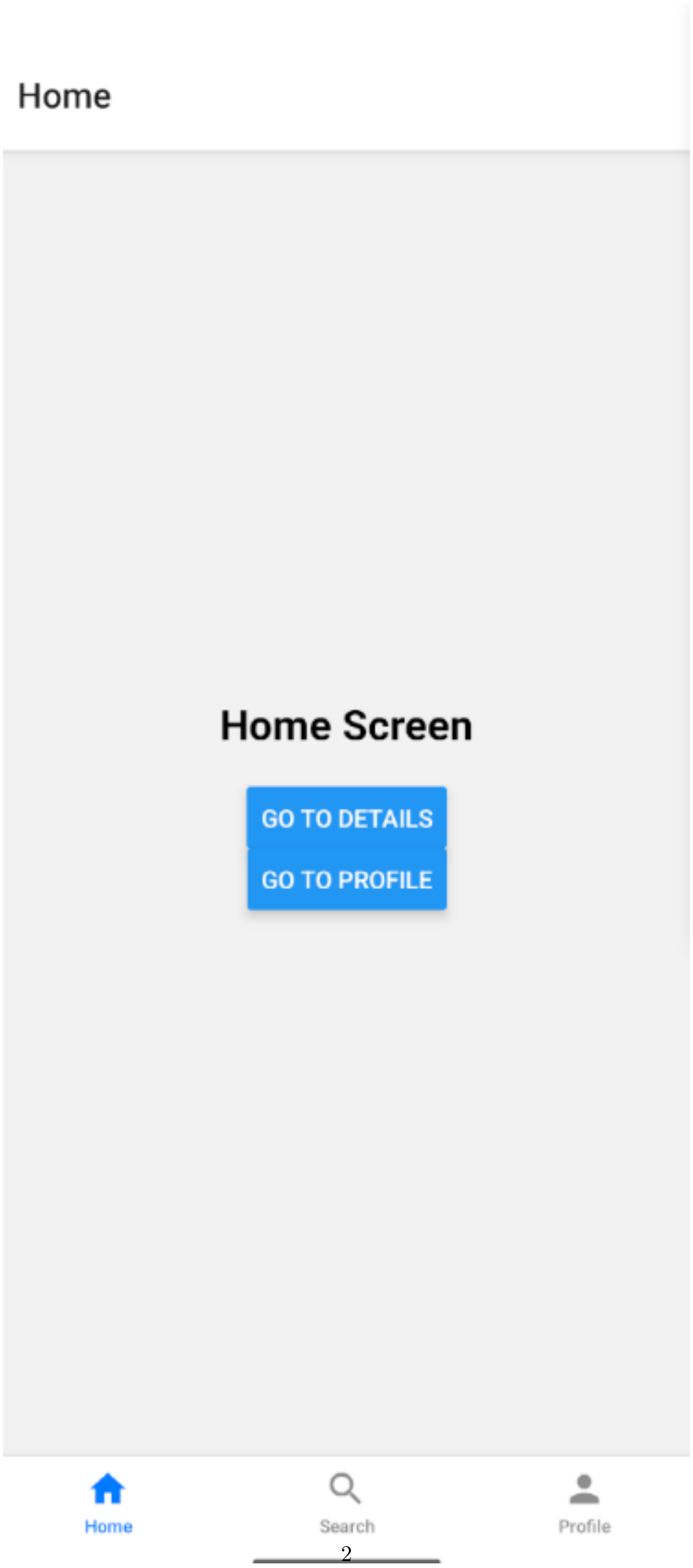


Figure 1: Home Screen with stack navigator

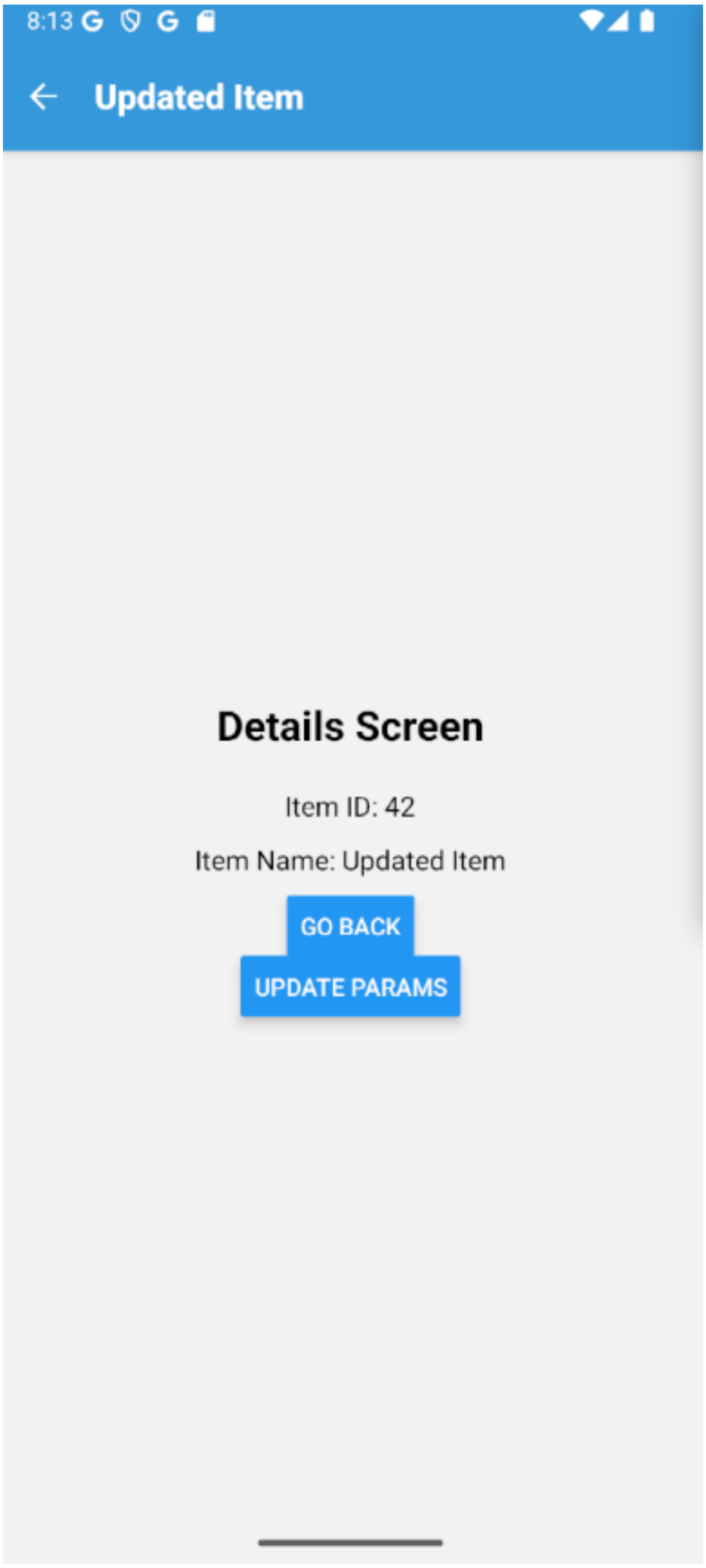
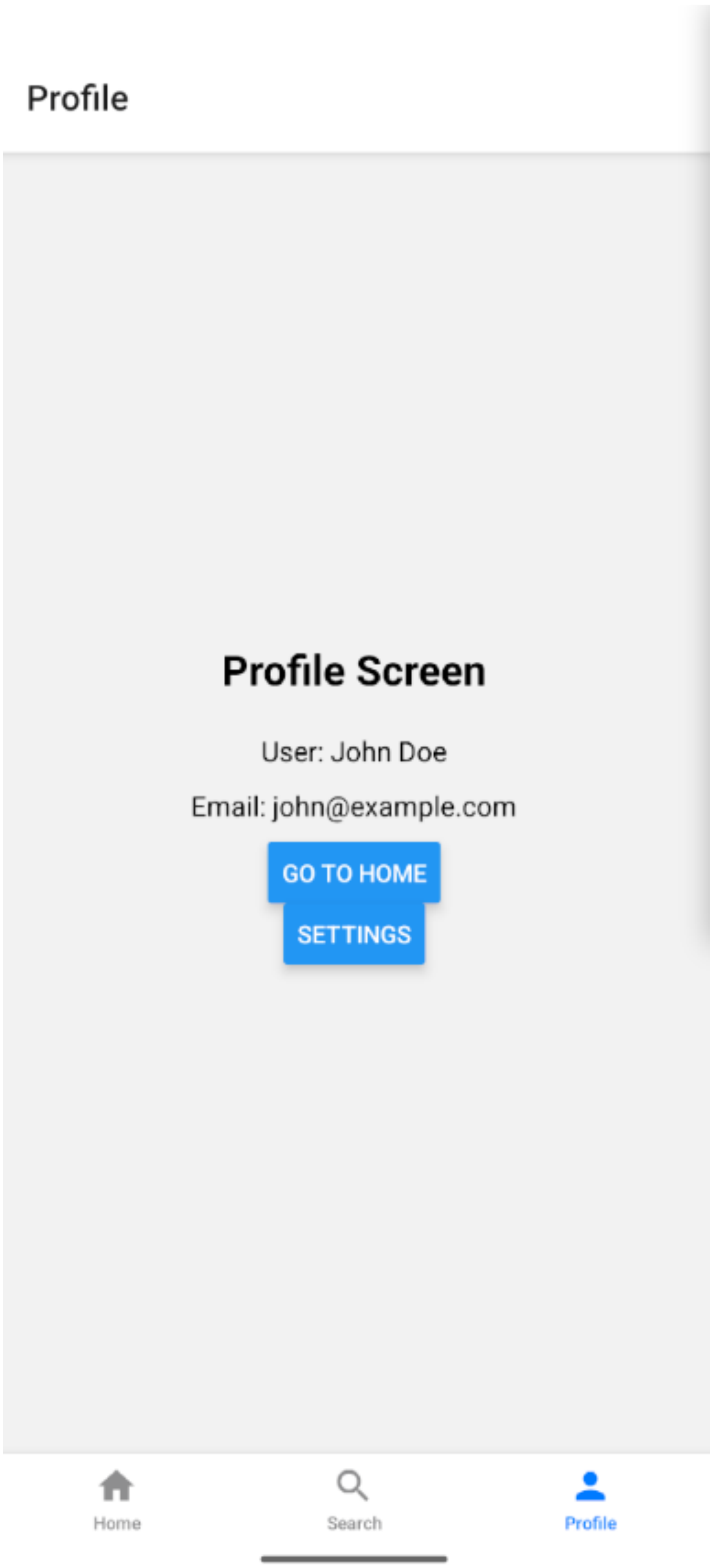


Figure 2: Details screen showing received parameters



User: John Doe

Email: john@example.com

Figure 3: Tab navigation showing all tabs

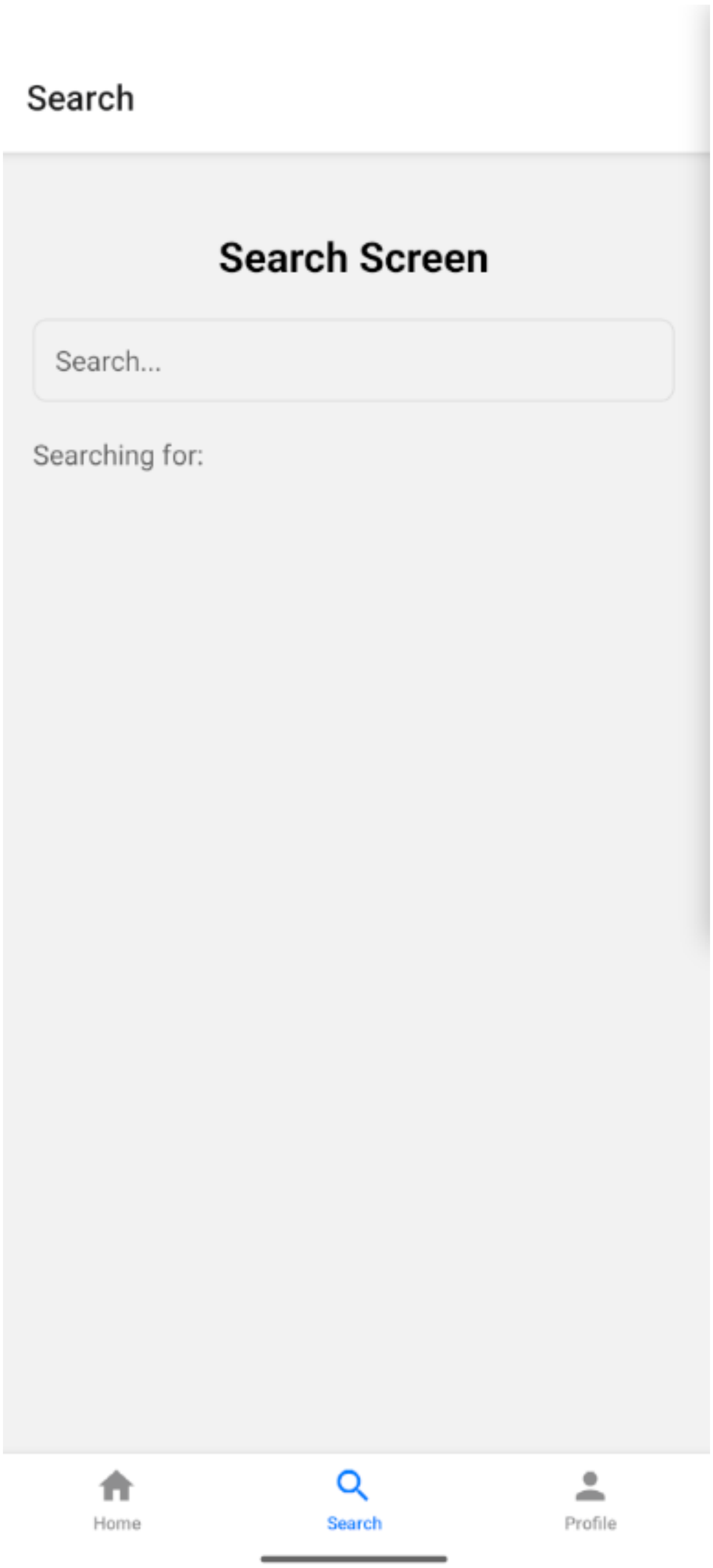
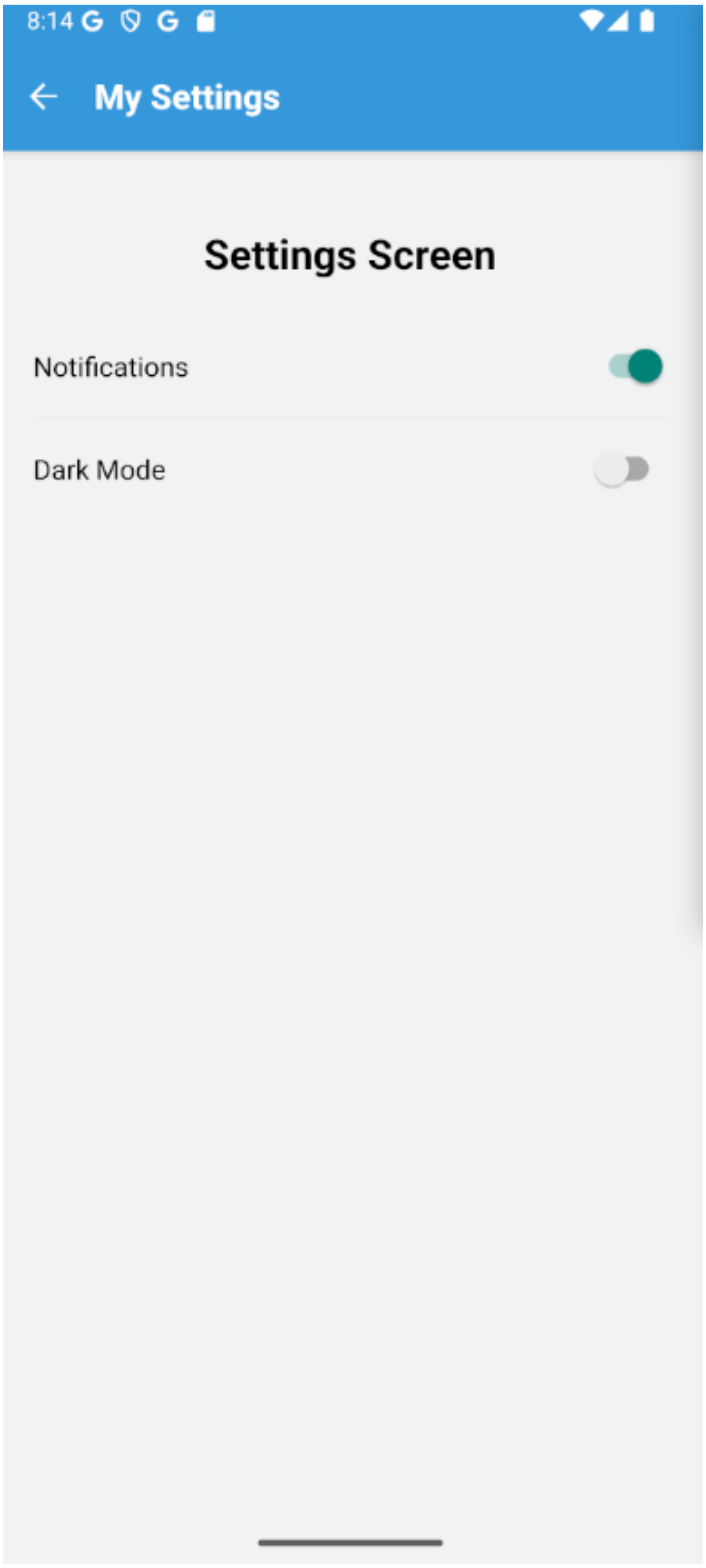


Figure 4: Search screen active



8:14

My Settings

Settings Screen

Notifications ●

Dark Mode ○

Figure 5: Settings screen with switches

2 Navigation Implementation

Navigation set up is structured with a Stack Navigation that has a nested Tab Navigation on the home page. To do this I had to set up the stack first.

```
const StackNavigator = () => {
        return (
                <Stack.Navigator
                        initialRouteName="Home"
                        screenOptions={{
                                headerStyle: {
                                        backgroundColor: '#3498db',
                                },
                                headerTintColor: '#fff',
                                headerTitleStyle: {
                                        fontWeight: 'bold',
                                },
                        }}>
                        <Stack.Screen
                                name="Home"
                                component={TabNavigator} //Nested Tab Navigation
                                options={{
                                        title: 'Home',
                                        headerShown: false,
                                }}
                        />
```

Figure 6: Segment of StackNavigation.js

In the above Figure 6 the *TabNavigator* is set as the component; effectively nesting the tab navigation within the *StackNavigation* component. Within Figure 1 you can see the home page and it's tabs, then navigate throughout the stack (seen in Figure 2).

2.1 Icons

Icon management had to be done with *@react-native-vector-icons/material-icons*

```
import { createBottomTabNavigator } from '@react-navigation/bottom-tabs';
import { MaterialIcons } from '@react-native-vector-icons/material-icons';

                                        ...

                <Tab.Navigator
                        screenOptions={({ route }) => ({
                                tabBarIcon: ({ focused, color, size }) => {
                                        let iconName;

                                        switch (route.name) {
                                                case 'Home':
                                                        iconName = 'home';
                                                        break;
                                                case 'Search':
                                                        iconName = 'search';
                                                        break;
                                                case 'Profile':
                                                        iconName = 'person';
                                                        break;
                                        }
                                        return <MaterialIcons name={iconName} size={size} color={col
                                },
                        })}
```

Figure 7: Segment of TabNavigator.js

In the above Figure 7, Icons using *MaterialIcons* are set up using a function that switches between icons based on the route then returns an icon component. Typically you need to handle passing paramters in nested components differently, but since we have all the primary navigation on the tab and links to things to the stack it wasn't really needed. If needed the figure below shows a snippet of the code.

```
<Button
        title="Go to Details"
        onPress={() =>
                navigation.navigate('Details', {
                        itemId: 42,
                        itemName: 'Sample Item',
                })
        }
/>
```

Figure 8: Segment of HomeScreen.js

3 Challenges and Learning

I had to nest the navigation since you can only have one root navigation as seen in the above figure 6. I also had to modify the manifest to opt-out of predicitive back on android since React Navigation doesn't yet support it.

```
android:enableOnBackInvokedCallback="false">
```

then I had to modify some code so *react-native-screens* can properly work

```
import android.os.Bundle
import com.swmansion.rnscreens.fragment.restoration.RNScreensFragmentFactory

...

  override fun onCreate(savedInstanceState: Bundle?) {
    supportFragmentManager.fragmentFactory = RNScreensFragmentFactory()
   SoLoader.init(this, false);
   super.onCreate(savedInstanceState)
```

Figure 9: Segment of MainActivity.ly

that was pretty much it. Oh, and updating to use *react-native-safe-area-context*, but I have covered that in previous labs.

4 Testing and Verification

I didn't need to test much since the only problem I had was the above mentioned and accidentally deleting part of the *AndroidManifest.xml* when opting-out of predicitive back; only took me about 2 hours to realize (not a brag). I did verify navigation passing by just seeing if the details page (in figure 2) actually loaded from the segment in figure 6.

5 diagram

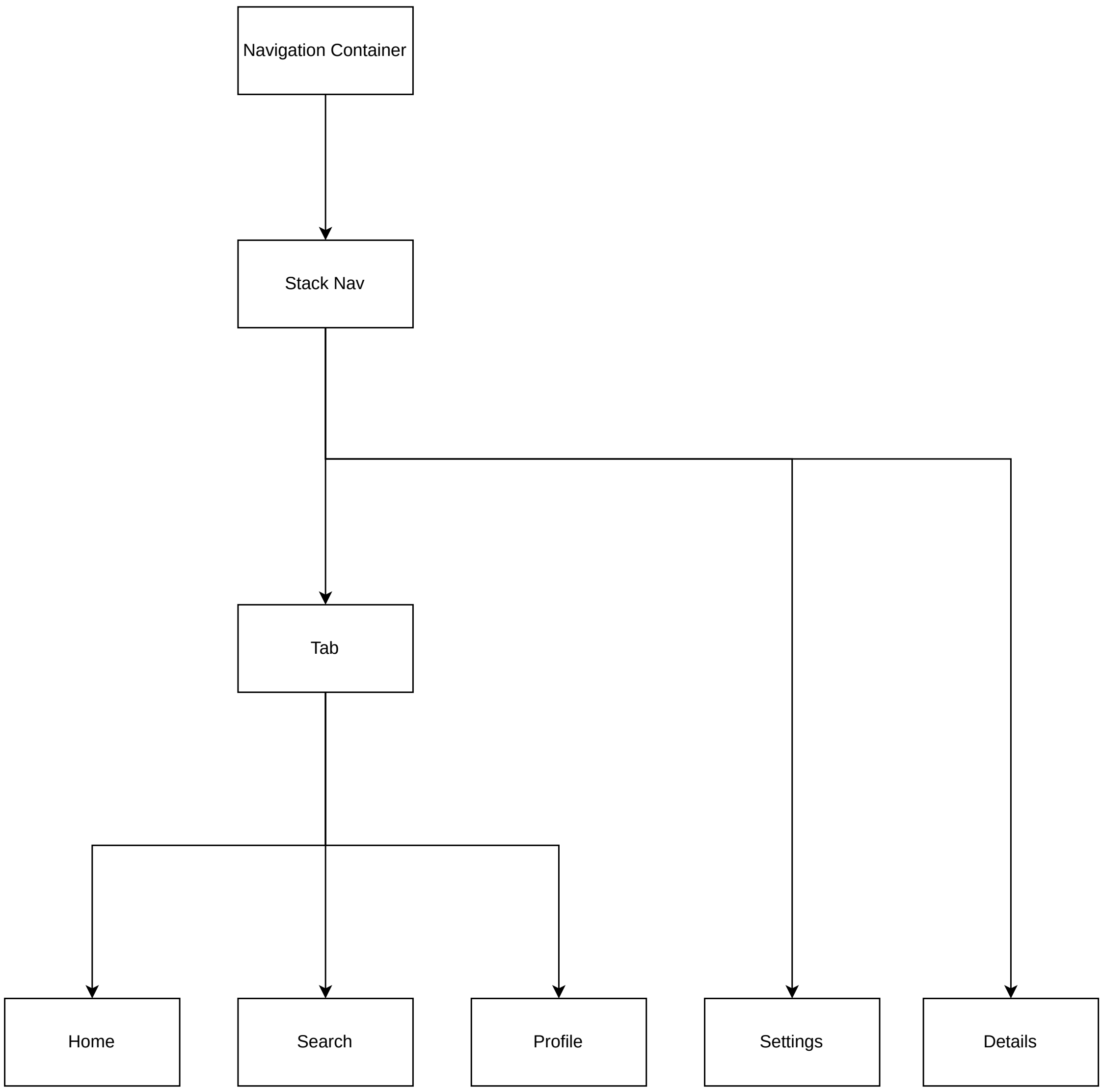


Figure 10: Navigation Flow