Open in app

Get started

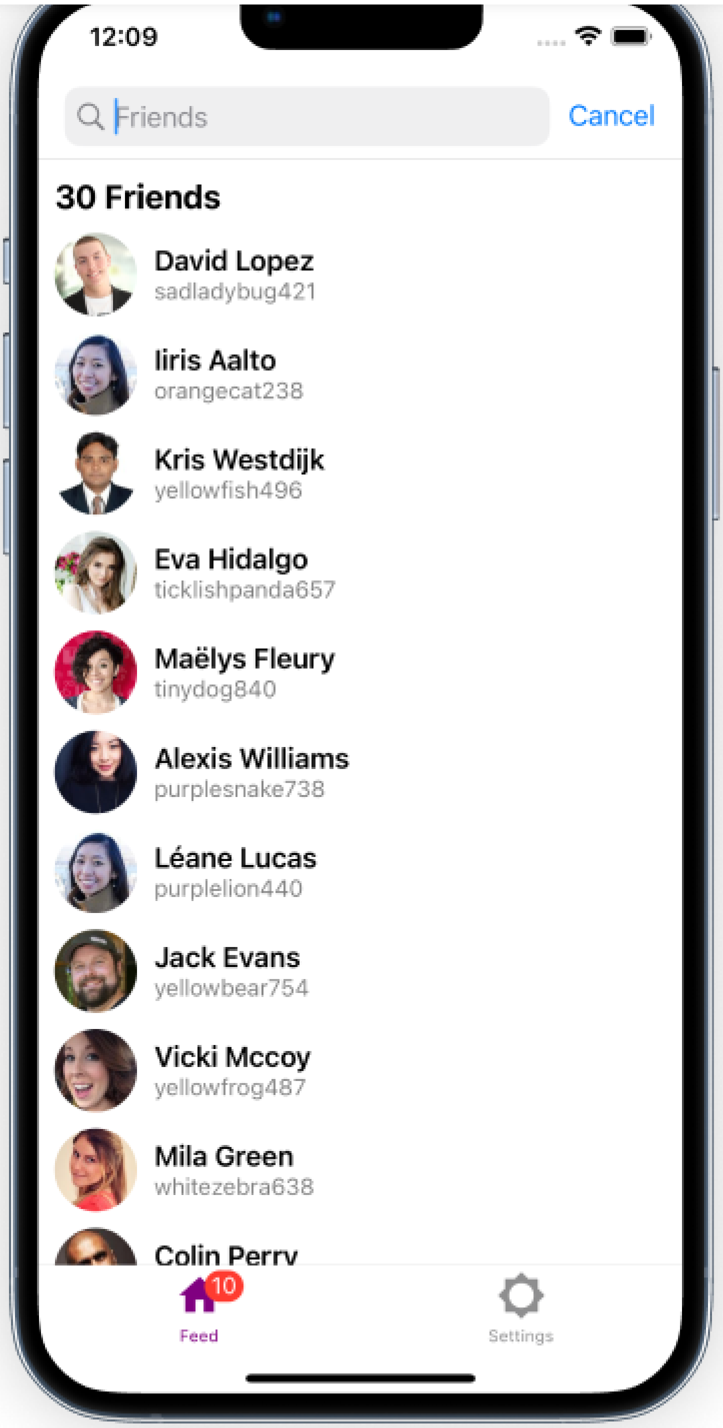
Betomoedano Follow

Dec 24, 2021 · 3 min read · Listen

Save

Search Filter React Native | Search Bar Tutorial

42



30 Friends

David Lopez
sadladybug421

Iiris Aalto
orangecat238

Kris Westdijk
yellowfish496

Eva Hidalgo
ticklishpanda657

Maëlys Fleury
tinydog840

Alexis Williams
purplesnake738

Léane Lucas
purplelion440

Jack Evans
yellowbear754

Vicki Mccoy
yellowfrog487

Mila Green
whitezebra638

Colin Perry

Hello everyone 👋

I am betomoedano and this is my first medium post ever!

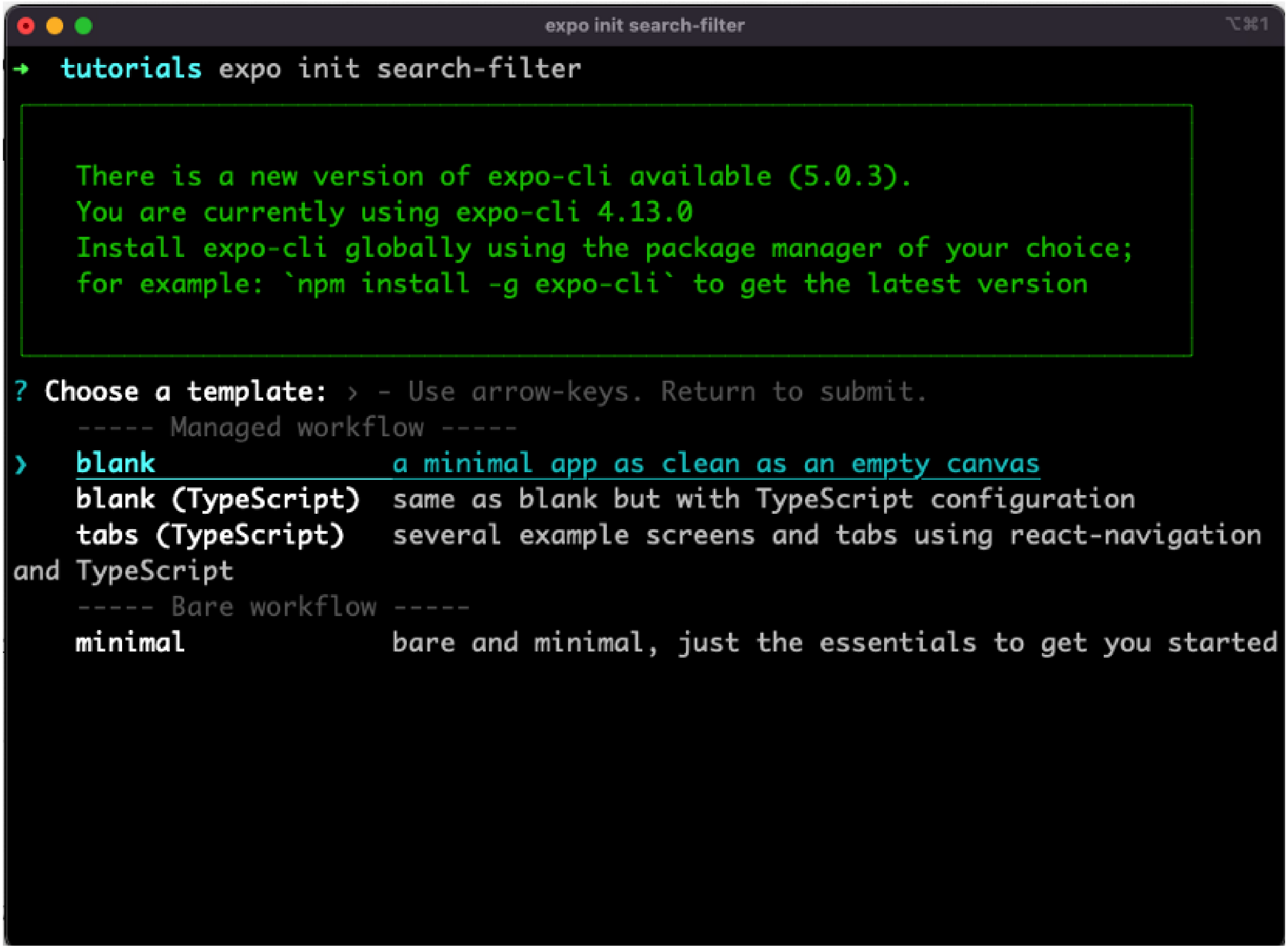
I am very excited to share some of the things that I have learned in my journey as a software engineer.

Creating the project

We are going to be using Expo for this project so the command to create our app would be something like this

#Create a project named search-filter

$ expo init search-filter



```
expo init search-filter
→ tutorials expo init search-filter

   There is a new version of expo-cli available (5.0.3).
   You are currently using expo-cli 4.13.0
   Install expo-cli globally using the package manager of your choice;
   for example: `npm install -g expo-cli` to get the latest version

? Choose a template: › - Use arrow-keys. Return to submit.
    ----- Managed workflow -----
❯   blank                 a minimal app as clean as an empty canvas
    blank (TypeScript)    same as blank but with TypeScript configuration
    tabs (TypeScript)     several example screens and tabs using react-navigation
and TypeScript
    ----- Bare workflow -----
    minimal               bare and minimal, just the essentials to get you started
```

Select blank for the template, wait for the dependencies to install then continue...

#Navigate to the project directory

$ cd search-filter

folder run the following commands.

$ yarn add @react-navigation/native

$ expo install react-native-screens react-native-safe-area-context

$ yarn add @react-navigation/native-stack

$ yarn add @react-navigation/bottom-tabs

Once the dependencies are installed we can go ahead and start de development server

# Start the development server

$ expo start

Navigation

Now we are going to set the navigation for our app

We have a bottom tab with two screens, *MyStack* and *Settings* and we also have a component called *MyStack* which contains the *Home* and the *Stack* screen

Here we have the code for the navigation.

```
import React from "react";
import { createBottomTabNavigator } from "@react-navigation/bottom-tabs";
import { createNativeStackNavigator } from "@react-navigation/native-stack";
import { NavigationContainer } from "@react-navigation/native";

//screens
import HomeScreen from "./screens/HomeScreen";
import SettingsScreen from "./screens/SettingsScreen";
import StackScreen from "./screens/StackScreen";

import MaterialCommunityIcons from 'react-native-vector-icons/MaterialCommunityIcons';
```

```
        <HomeStackNavigator.Navigator
            initialRouteName="HomeScreen"
        >
            <HomeStackNavigator.Screen
                name="HomeScreen"
                component={HomeScreen}
            />
            <HomeStackNavigator.Screen
                name="Stack"
                component={StackScreen}
                options={{
                    headerBackTitleVisible: false,
                }}
            />
        </HomeStackNavigator.Navigator>
    )
}

const Tab = createBottomTabNavigator();

function MyTabs() {
  return (
      <Tab.Navigator
        initialRouteName="Home"
        screenOptions= {{
            tabBarActiveTintColor: 'purple',
        }}
      >
        <Tab.Screen
            name="Home"
            component={MyStack}
            options={{
                tabBarLabel: 'Feed',
                tabBarIcon: ({ color, size }) => (
                    <MaterialCommunityIcons name="home" color={color} size={30} />
                ),
                tabBarBadge: 10,
                headerShown: false,
            }}
        />
        <Tab.Screen
            name="Settings"
                component={SettingsScreen}
                ),
            }}
        />
    </Tab.Navigator>
    );
}

export default function Navigation() {
  return (
    <NavigationContainer>
      <MyTabs />
    </NavigationContainer>
  );
```

Note that we also need to create our screens files:

”HomeScreen.js”

“SettingsScreen.js”

“StackScreen.js”

For SettingsScreen and StackScreen we will just show a text as the following code.

```
import React from "react";
import { View, Text } from "react-native";

const SettingsScreen = () => {
    return (
        <View>
            <Text
                style={{
                    fontSize: 30,
                    textAlign: "center",
                    marginTop: "20%"
                }}
            >Settings Screen</Text>
        </View>
    );
    }

```

```
import React from "react";
import { View, Text } from "react-native";

const StackScreen = () => {
    return (
        <View>
            <Text
                style={{
                    fontSize: 30,
                    textAlign: "center",
                    marginTop: "20%"
                }}
            >Stack Screen</Text>
        </View>
    );
    }

export default StackScreen;
```

StackScreen.js hosted with ❤ by GitHub view raw

Code for Stack Screen

Finally we can start working on our HomeScreen.js file, for now we can just show a text as well while we are working on getting the fake data from our API.

```
import React from "react";
import { View, Text } from "react-native";

const HomeScreen = () => {
    return (
        <View>
            <Text
                style={{
                    fontSize: 30,
                    textAlign: "center",
                    marginTop: "20%"
                }}
            >HomeScreen Screen</Text>
        </View>
    );
    }
```

Getting fake data from Random user API

After we have our project running we can now get our fake data from our API.

We will need to import useEffect and useState from react, we create a variable called ‘data’ that is going to contain the fake users, then we simply use the built-in function “fetch” to get the data then we transform the response to a json file and finally, we set our data.

You can console.log the response to check what kind of data we got and play around with it.

```
import React, { useEffect, useState } from "react";
import { View, Text } from "react-native";

const HomeScreen = () => {

    const [data, setData] = useState([]);

    useEffect(() => {
        fetchData("https://randomuser.me/api/?results=20");
    }, []);

    const fetchData = async (url) => {
        try {
            const response = await fetch(url);
            const json = await response.json();
            setData(json.results);
            setFilteredData(json.results);
            console.log(json.results);
        } catch (error) {
            console.error(error);
        }
    };

    return (
        <View>
          <Text
                style={{
                    fontSize: 30,
                    textAlign: "center",
    );
  }

export default HomeScreen;
```

Displaying the data

Now that we have our data, we need to show it on screen.

We will map throw the data array and render a simple component that shows each user in our array.

```
    return (
        <ScrollView>
            <Text style={styles.textFriends}>{data.length} Friends</Text>
            {
                data.map((item, index) => {
                    return (
                        <View key={index} style={styles.itemContainer}>
                            <Image
                                source={{ uri: item.picture.large }}
                                style={styles.image}
                            />
                            <View>
                                <Text style={styles.textName}>{item.name.first} {item.name.last
                                <Text style={styles.textEmail}>{item.login.username}</Text>
                            </View>
                        </View>
                    )
                })
            }
        </ScrollView>
    );
    }

export default HomeScreen;

const styles = StyleSheet.create({
    textFriends: {
        fontSize: 20,
        textAlign: 'left',
        flexDirection: 'row',
        alignItems: 'center',
        marginLeft: 10,
        marginTop: 10,
    },
    image: {
        width: 50,
        height: 50,
        borderRadius: 25,
    },
    textName: {
        fontSize: 17,
        marginLeft: 10,
        fontWeight: "600",
    },
    textEmail: {
        fontSize: 14,
        marginLeft: 10,
        color: "grey",
    },
});
```

Adding the search bar and filtering data. Finally! 🎉

1. - First, we need to import *"useNavigation"*

2. - Using another **useEffect** we will set the header options for **HomeScreen**

3. - We also need another variable to hold the filtered data

4. - Finally, we create a function called **searchFilterFunction()** that will check if we have text in the search bar, if we have text then we will pass that text to uppercase and since we are filtering the data by name we also pass the name to uppercase. Then we simply return the filtered data using the method **indexOf()** which returns the first index at which a given element (text) can be found in the array, or -1 if it is not present.

After we add that our code should look like this.

```
import React, {useEffect, useState} from "react";
```

```
    const navigation = useNavigation();
    const [data, setData] = useState([]);
    const [filteredData, setFilteredData] = useState([]);

    useEffect(() => {
        fetchData("https://randomuser.me/api/?results=30");
    }, []);

    useEffect(() => {
        navigation.setOptions({
            headerLargeTitle: true,
            headerTitle: "Home",
            headerRight: () => (
                <TouchableOpacity
                    onPress={() => navigation.navigate("Stack")}
                    style={{
                        backgroundColor: "purple",
                        width: 30,
                        height: 30,
                        borderRadius: 10,
                        justifyContent: "center",
                    }}
                >
                    <Text
                        style={{
                            fontSize: 20,
                            textAlign: "center",
                            color: "white",
                        }}
                    >+</Text>
                </TouchableOpacity>
            ),
            headerSearchBarOptions: {
                placeholder: "Friends",
                onChangeText: (event) => {
                    searchFilterFunction(event.nativeEvent.text);
                },
            },
        });
    }, [navigation]);

    const fetchData = async (url) => {
        try {
```

```
        console.log(json.results);
      } catch (error) {
        console.error(error);
      }
    };

    const searchFilterFunction = (text) => {
      if(text){
        const newData = data.filter(item => {
          const itemData = item.name.first ? item.name.first.toUpperCase() : ''.toUpperC
          const textData = text.toUpperCase();
          return itemData.indexOf(textData) > -1;
        })
        setFilteredData(newData);
      } else {
        setFilteredData(data);
      }
    }

    return (
      <ScrollView>
        <Text style={styles.textFriends}>{filteredData.length} Friends</Text>
        {
          filteredData.map((item, index) => {
            return (
              <View key={index} style={styles.itemContainer}>
                <Image
                  source={{ uri: item.picture.large }}
                  style={styles.image}
                />
                <View>
                  <Text style={styles.textName}>{item.name.first} {item.name.las
                  <Text style={styles.textEmail}>{item.login.username}</Text>
                </View>
              </View>
            )
          })
        }
      </ScrollView>
    );
  }

export default HomeScreen;
```

```
        marginLeft: 10,
        fontWeight: 'bold',
        marginTop: 10,
    },
    itemContainer: {
        flexDirection: 'row',
        alignItems: 'center',
        marginLeft: 10,
        marginTop: 10,
    },
    image: {
        width: 50,
        height: 50,
        borderRadius: 25,
    },
    textName: {
        fontSize: 17,
        marginLeft: 10,
        fontWeight: "600",
    },
    textEmail: {
        fontSize: 14,
        marginLeft: 10,
        color: "grey",
```

Conclusion

That's all it takes to create that useful functionality. Happy Coding!

Buy me a coffee! ☕