



## Exercise (Instructions): Picking an Image

### Objectives and Outcomes

In this exercise you will use the ImagePicker API from Expo SDK to access the camera to obtain a picture for use in our app. At the end of this exercise you will be able to:

- Configure your app to use the ImagePicker
- Obtain a picture from the Camera using the ImagePicker

### Obtaining a Picture from the Camera

- Open LoginComponent.js and update its content as follows:

```
1 import React, { Component } from 'react';
2 import { View, StyleSheet, Text, ScrollView, Image } from 'react-native';
3 import { Input, CheckBox, Button, Icon } from 'react-native-elements';
4 import { SecureStore, Permissions, ImagePicker } from 'expo';
5 import { createBottomTabNavigator } from 'react-navigation';
6 import { baseUrl } from '../shared/baseUrl';
7
8 class LoginTab extends Component {
9
10   constructor(props) {
11     super(props);
12
13     this.state = {
14       username: '',
15       password: '',
16       remember: false
17     }
18   }
19
20   componentDidMount() {
21     SecureStore.getItemAsync('userinfo')
22       .then(userdata) => {
23         let userinfo = JSON.parse(userdata);
24         if (userinfo) {
25           this.setState({username: userinfo.username});
26           this.setState({password: userinfo.password});
27           this.setState({remember: true})
28         }
29       }
30   }
31
32   static navigationOptions = {
33     title: 'Login',
34     tabBarIcon: ({ tintColor }) => (
35       <Icon
36         name='sign-in'
37         type='font-awesome'
38         size={24}
39         iconStyle={{ color: tintColor }}
40       />
41     )
42   };
43
44   handleLogin() {
45     console.log(JSON.stringify(this.state));
46     if (this.state.remember)
47       SecureStore.setItemAsync('userinfo', JSON.stringify({username: this
48         .state.username, password: this.state.password}))
49       .catch((error) => console.log('Could not save user info', error
50         ));
51     else
52       SecureStore.deleteItemAsync('userinfo')
53       .catch((error) => console.log('Could not delete user info',
54         error));
55   }
56
57   render() {
58     return (
59       <View style={styles.container}>
60         <Input
61           placeholder="Username"
62           leftIcon={{ type: 'font-awesome', name: 'user-o' }}
63           onChangeText={(username) => this.setState({username})}
64           value={this.state.username}
65           containerStyle={styles.formInput}
66         />
67         <Input
68           placeholder="Password"
69           leftIcon={{ type: 'font-awesome', name: 'key' }}
70           onChangeText={(password) => this.setState({password})}
71           value={this.state.password}
72           containerStyle={styles.formInput}
73         />
74         <CheckBox title="Remember Me"
75           checked={this.state.remember}
76           onPress={() => this.setState({remember: !this.state
77             .remember})}
78           containerStyle={styles.formCheckbox}
79         />
80         <View style={styles.formButton}>
81           <Button
82             onPress={() => this.handleLogin()}
83             title="Login"
84             icon={
85               <Icon
86                 name='sign-in'
87                 type='font-awesome'
88                 size={24}
89                 color= 'white'
90               />
91             }
92             buttonStyle={{
93               backgroundColor: "#512DA8"
94             }}
95           />
96         </View>
97         <View style={styles.formButton}>
98           <Button
99             onPress={() => this.props.navigation.navigate("Register")
100             }
101             title="Register"
102             clear
103             icon={
104               <Icon
105                 name='user-plus'
106                 type='font-awesome'
107                 size={24}
108                 color= 'blue'
109               />
110             }
111             titleStyle={{
112               color: "blue"
113             }}
114           />
115         </View>
116       </View>
117     );
118   }
119 }
120
121 class RegisterTab extends Component {
```

```

121 - constructor(props) {
122 -   super(props);
123 -
124 -   this.state = {
125 -     username: '',
126 -     password: '',
127 -     firstname: '',
128 -     lastname: '',
129 -     email: '',
130 -     remember: false,
131 -     imageUrl: baseUrl + 'images/logo.png'
132 -   }
133 - }
134 -
135 - getImageFromCamera = async () => {
136 -   const cameraPermission = await Permissions.askAsync(Permissions.CAMERA);
137 -   const cameraRollPermission = await Permissions.askAsync(Permissions
138 -     .CAMERA_ROLL);
139 -
140 -   if (cameraPermission.status === 'granted' && cameraRollPermission.status
141 -     === 'granted') {
142 -     let capturedImage = await ImagePicker.launchCameraAsync({
143 -       allowsEditing: true,
144 -       aspect: [4, 3],
145 -     });
146 -     if (!capturedImage.cancelled) {
147 -       console.log(capturedImage);
148 -       this.setState({imageUrl: capturedImage.uri });
149 -     }
150 -   }
151 - }
152 -
153 - static navigationOptions = {
154 -   title: 'Register',
155 -   tabBarIcon: ({ tintColors, focused }) => (
156 -     <Icon
157 -       name='user-plus'
158 -       type='font-awesome'
159 -       size={24}
160 -       iconStyle={{ color: tintColors }}
161 -     />
162 -   );
163 -
164 -   handleRegister() {
165 -     console.log(JSON.stringify(this.state));
166 -     if (this.state.remember)
167 -       SecureStore.setItemAsync('userinfo', JSON.stringify({username: this
168 -         .state.username, password: this.state.password}))
169 -         .catch((error) => console.log('Could not save user info', error
170 -           ));
171 -   }
172 -
173 -   render() {
174 -     return(
175 -       <ScrollView>
176 -         <View style={styles.container}>
177 -           <Image
178 -             source={{uri: this.state.imageUrl}}
179 -             loadingIndicatorSource={require('./images/logo.png')}
180 -             style={styles.image}
181 -           />
182 -           <Button
183 -             title="Camera"
184 -             onPress={this.getImageFromCamera}
185 -           />
186 -         </View>
187 -         <Input
188 -           placeholder="Username"
189 -           leftIcon={{ type: 'font-awesome', name: 'user-o' }}
190 -           onChangeText={(username) => this.setState({username})}
191 -           value={this.state.username}
192 -           containerStyle={styles.formInput}
193 -         />
194 -         <Input
195 -           placeholder="Password"
196 -           leftIcon={{ type: 'font-awesome', name: 'key' }}
197 -           onChangeText={(password) => this.setState({password})}
198 -           value={this.state.password}
199 -           containerStyle={styles.formInput}
200 -         />
201 -         <Input
202 -           placeholder="First Name"
203 -           leftIcon={{ type: 'font-awesome', name: 'user-o' }}
204 -           onChangeText={(firstname) => this.setState({firstname})}
205 -           value={this.state.firstname}
206 -           containerStyle={styles.formInput}
207 -         />
208 -         <Input
209 -           placeholder="Last Name"
210 -           leftIcon={{ type: 'font-awesome', name: 'user-o' }}
211 -           onChangeText={(lastname) => this.setState({lastname})}
212 -           value={this.state.lastname}
213 -           containerStyle={styles.formInput}
214 -         />
215 -         <Input
216 -           placeholder="Email"
217 -           leftIcon={{ type: 'font-awesome', name: 'envelope-o' }}
218 -           onChangeText={(email) => this.setState({email})}
219 -           value={this.state.email}
220 -           containerStyle={styles.formInput}
221 -         />
222 -         <CheckBox title="Remember Me"
223 -           center
224 -           checked={this.state.remember}
225 -           onPress={() => this.setState({remember: !this.state
226 -             .remember})}
227 -           containerStyle={styles.formCheckbox}
228 -         />
229 -         <View style={styles.formButton}>
230 -           <Button
231 -             onPress={() => this.handleRegister()}
232 -             title="Register"
233 -             icon={
234 -               <Icon
235 -                 name='user-plus'
236 -                 type='font-awesome'
237 -                 size={24}
238 -                 color='white'
239 -               />
240 -             }
241 -             buttonStyle={{
242 -               backgroundColor: "#512DA8"
243 -             }}
244 -           />
245 -         </View>
246 -       </ScrollView>
247 -     );
248 -   }
249 - }
250 -
251 - const styles = StyleSheet.create({
252 -   container: {
253 -     justifyContent: 'center',
254 -     margin: 20,
255 -   },
256 -   imageContainer: {
257 -     flex: 1,
258 -     flexDirection: 'row',
259 -     margin: 20
260 -   },
261 -   image: {
262 -     margin: 10,
263 -     width: 80,
264 -     height: 60
265 -   },
266 -   formInput: {
267 -     margin: 20
268 -   },
269 -   formCheckbox: {
270 -     margin: 20,
271 -     backgroundColor: null
272 -   },
273 -   formButton: {
274 -     margin: 60
275 -   }
276 - });

```

```
274     }
275   });
276
277   const Login = createBottomTabNavigator({
278     Login: LoginTab,
279     Register: RegisterTab
280   }, {
281     tabBarOptions: {
282       activeBackgroundColor: '#9575CD',
283       inactiveBackgroundColor: '#D1C4E9',
284       activeTintColor: 'ffffff',
285       inactiveTintColor: 'gray',
286     }
287   });
288
289   export default Login;
```

- Then, open MainComponent.js and update it as follows:

```
1  - - -
2
3  const LoginNavigator = createStackNavigator({
4    Login: Login
5  }, {
6    navigationOptions: ({ navigation }) => ({
7      headerStyle: {
8        backgroundColor: "#512DA8"
9      },
10     headerTitleStyle: {
11       color: "ffff"
12     },
13     title: 'Login',
14     headerTintColor: "ffff",
15     headerLeft: <icon name="menu" size={24}
16       iconStyle={{ color: 'white' }}
17     onPress={() => navigation.toggleDrawer()} />
18   })
19 });
20
21 - - -
```

- Save all the changes and do a Git commit with the message "Picking an Image".

## Conclusions

In this exercise we use the ImagePicker API to obtain an image from the camera.

Mark as completed