

```
1 //Librairies
2 import React from 'react';
3 import { NavigationContainer } from '@react-navigation/native';
4 //Redux et Persist
5 import { Provider } from 'react-redux';
6 import { persistStore } from 'redux-persist';
7 import { PersistGate } from 'redux-persist/es/integration/react';
8 import Store from './Store/configureStore.js';
9 //Components perso
10 import Index from './Components/Index.js';
11
12 class App extends React.Component {
13   render() {
14     let persistor = persistStore(Store);
15     return (
16       <Provider store={Store}>
17         <PersistGate persistor={persistor}>
18           <NavigationContainer>
19             <Index/>
20           </NavigationContainer>
21         </PersistGate>
22       </Provider>
23     )
24   }
25 }
26
27 //Export du component
28 export default App;
```

```
1 //Librairies
2 import { createStore } from 'redux';
3 import { persistCombineReducers } from 'redux-persist';
4 import AsyncStorage from '@react-native-community/async-storage';
5 //Components perso
6 import isConnected from './Reducers/connected.js';
7
8 //Configuration
9 const rootPersistConfig = {
10   key: 'root',
11   storage: AsyncStorage
12 }
13
14 //Export du component
15 export default createStore(persistCombineReducers(rootPersistConfig,
  {isConnected}))
```

```
1 //State initial
2 const initialState = { idUser: false }
3
4 //Connecter ou déconnecter l'utilisateur
5 function isConnected(state = initialState, action) {
6   let nextState;
7   switch (action.type) {
8     case 'IS_CONNECTED':
9       //Déconnecter l'user
10      if(action.value === false){
11        nextState = {
12          ...state,
13          idUser: false
14        }
15      }else{
16        //Connecter l'utilisateur
17        nextState = {
18          ...state,
19          idUser: action.value
20        }
21      }
22      return nextState || state
23    default:
24      return state
25  }
26 }
27
28 //Export du component
29 export default isConnected
```

```
1 //Librairies
2 import React, { Component } from 'react';
3 import { View } from 'react-native';
4 //Components perso
5 import { BASE_STYLE, Loading, ImageList } from './MyComponent.js';
6 //Fonctions
7 import { getFav } from '../WS/functions.js';
8 //Redux
9 import { connect } from 'react-redux';
10
11 class Fav extends Component {
12   constructor(props) {
13     super(props);
14     //State
15     this.state = {
16       wallpapers: undefined,
17       isLoading: true,
18       refreshing: false
19     };
20     //Récupérer les favoris
21     getFav(this.props.idUser).then(data => this.setState({
22       wallpapers: data,
23       isLoading: false
24     }));
25   }
26
27   //Quand la liste est rafraîchit
28   _onRefresh = () => {
29     this.setState({refreshing: true});
30     //Récupérer les favoris
31     getFav(this.props.idUser).then(data => this.setState({
32       wallpapers: data,
33       refreshing: false
34     }));
35   }
36
37   render(){
38     return (
39       <View style={BASE_STYLE.container}>
40         {/* Affichage des fonds d'écrans */}
41         <ImageList
42           data={this.state.wallpapers}
43           onPress={(item) =>
44             this.props.navigation.navigate('WallpaperDetails', {wallpaper: item})}
45           refreshing={this.state.refreshing}
46           onRefresh={() => this._onRefresh()}
47         />
48         {/* Affichage du chargement */}
49         {this.state.isLoading && <Loading/>}
50       </View>
51     );
52   }
53 }
54 //Pour Redux
```

```
55 const mapStateToProps = (state) => {  
56   return {  
57     idUser: state.isConnected.idUser  
58   }  
59 }  
60  
61 //Export du component  
62 export default connect(mapStateToProps)(Fav);
```

```
1 //Librairies
2 import { Dimensions } from 'react-native';
3 import { createStackNavigator } from '@react-navigation/stack';
4 import { createBottomTabNavigator } from '@react-navigation/bottom-tabs';
5
6 //Constantes
7
8 //Liens
9 const URL_WS = 'https://flobeney.ch/wpheaven/ws/';
10 const URL_API = 'https://wallhaven.cc/api/v1/';
11
12 //Dimensions de l'appareil
13 export const HEIGHT_SCREEN = Dimensions.get("screen").height;
14 export const WIDTH_SCREEN = Dimensions.get("screen").width;
15 const PGCD_SCREEN_DIMENSIONS = pgcd(HEIGHT_SCREEN, WIDTH_SCREEN);
16 //Ratio de l'appareil
17 const RATIO = (WIDTH_SCREEN / PGCD_SCREEN_DIMENSIONS).toString() + 'x' +
  (HEIGHT_SCREEN / PGCD_SCREEN_DIMENSIONS).toString();
18
19 //Constantes couleurs des onglets lors du focus ou non
20 export const COLOR_NOT_FOCUSED = '#b7b7b7';
21 export const COLOR_FOCUSED = '#000';
22
23 //Messages d'erreur
24 export const TEXT_FILL_ALL = 'Please fill in all values';
25
26 //Navigateur "Stack" (écran qui peuvent s'empiler)
27 export const Stack = createStackNavigator();
28 //Navigateur "Tab" (onglet en bas de l'application)
29 export const Tab = createBottomTabNavigator();
30
31 // Données de recherches
32 const CATEGORY_GENERAL = "General";
33 const CATEGORY_ANIME = "Anime";
34 const CATEGORY_PEOPLE = "People";
35 export const CATEGORIES = [
36   {
37     name: CATEGORY_GENERAL,
38     value: 0
39   },
40   {
41     name: CATEGORY_ANIME,
42     value: 0
43   },
44   {
45     name: CATEGORY_PEOPLE,
46     value: 0
47   }
48 ];
49
50 const SORTING_DATE = 'date_added';
51 const SORTING_RELEVANCE = 'relevance';
52 const SORTING_RANDOM = 'random';
53 const SORTING_VIEWS = 'views';
54 const SORTING_FAVORITES = 'favorites';
```

```
54 const SORTING_FAVORITES = 'favorites',
55 const SORTING_TOPLIST = 'toplist';
56 export const SORTING = [
57   SORTING_DATE,
58   SORTING_RELEVANCE,
59   SORTING_RANDOM,
60   SORTING_VIEWS,
61   SORTING_FAVORITES,
62   SORTING_TOPLIST
63 ];
64 export const ORDER_DESC = 'desc';
65 export const ORDER_ASC = 'asc';
66
67 //Fonctions
68
69 //Calcul du PGCD
70 function pgcd(nbOne, nbTwo){
71   //Si le 2ème chiffre est 0 (division impossible)
72   if(nbTwo == 0){
73     return Math.abs(nbOne);
74   }
75   var temp = nbOne % nbTwo;
76   if(temp == 0){
77     //Un pgcd n'a pas de signe
78     return Math.abs(nbTwo);
79   }
80   return pgcd(nbTwo, temp);
81 }
82
83 //Vérifier que tout les éléments d'un objet sont différents de null
84 export function nothingEmpty(obj){
85   var res = true;
86   Object.values(obj).forEach(function(item){
87     if(item == ''){
88       res = false;
89     }
90   })
91
92   return res;
93 }
94
95 //Trimer toutes les valeurs d'un objet
96 export function trimValues(obj){
97   const res = Object.fromEntries(
98     Object.entries(obj)
99     .map(([ key, val ]) => [ key, val.trim() ])
100   );
101   return res;
102 }
103
104 //Wallhaven
105
106 //Récupérer les derniers fonds d'écran en fonction du ratio de l'écran de
//l'utilisateur
107 export function getLastWP(){
108   //lien
```

```
108 //Lien
109 var link = URL_API + 'search?ratios=' + RATIO;
110 //Appel fetch
111 return fetch(link, {
112     method: 'GET',
113     headers: {
114         Accept: 'application/json', //type de retour
115     }
116 })
117 .then((response) => response.json())
118 .catch((error) => console.error(error));
119 }
120
121 //Récupérer les fonds d'écran similaires en fonction du ratio de l'écran de
l'utilisateur
122 export function getSimilarWP(id){
123     //Lien
124     var link = URL_API + 'search?q=like:' + id + '&ratios=' + RATIO;
125     //Appel fetch
126     return fetch(link, {
127         method: 'GET',
128         headers: {
129             Accept: 'application/json', //type de retour
130         }
131     })
132     .then((response) => response.json())
133     .catch((error) => console.error(error));
134 }
135
136 //Recherche de fond d'écran
137 export function searchWP(params, page = 1){
138     //lien
139     var link = setAPISearchLink(params, page);
140     //Appel fetch
141     return fetch(link, {
142         method: 'GET',
143         headers: {
144             Accept: 'application/json', //type de retour
145         }
146     })
147     .then((response) => response.json())
148     .catch((error) => console.error(error));
149 }
150
151 //Return the link of the search
152 function setAPISearchLink(params, page){
153     var link = URL_API + 'search?page=' + page;
154     if(params.color != null){
155         link += '&colors=' + params.color;
156     }
157     if(params.category != null){
158         link += '&categories=' + params.category;
159     }
160     if(params.order != null){
161         link+= '&order='+ params.order;
162     }
163 }
```



```
162     },
163     if(params.sorting != null){
164         link += '&sorting=' + params.sorting;
165     }
166     if(params.owner != null || params.tag != null){
167         link += '&q=';
168         if(params.tag != null){
169             link += params.tag;
170         }
171         if(params.owner != null){
172             link += '@' + params.owner;
173         }
174     }
175     return link + '&ratios=' + RATIO;
176 }
177
178 //SQL
179
180 //Utilisateur (connexion, inscription)
181
182 //Connexion
183 export function login(user){
184     //Lien
185     var link = URL_WS + 'login.php';
186     //FormData
187     var formData = new FormData();
188     //Ajouter les données
189     formData.append('pseudo', user.pseudo);
190     formData.append('pwd', user.pwd);
191     //Appel fetch
192     return fetch(link, {
193         method: 'POST',
194         headers: {
195             Accept: 'application/json', //type de retour
196             'Content-Type': 'multipart/form-data' //type d'envoi
197         },
198         body: formData, //Requete
199     })
200     .then((response) => response.json())
201     .catch((error) => console.error(error));
202 }
203
204 //Inscription
205 export function signUp(user){
206     //Lien
207     var link = URL_WS + 'signUp.php';
208     //FormData
209     var formData = new FormData();
210     //Ajouter les données
211     formData.append('pseudo', user.pseudo);
212     formData.append('email', user.email);
213     formData.append('pwd', user.pwd);
214     //Appel fetch
215     return fetch(link, {
216         method: 'POST',
```

```
217     headers: {
218         Accept: 'application/json', //type de retour
219         'Content-Type': 'multipart/form-data' //type d'envoi
220     },
221     body: formData, //Requete
222 })
223 .then((response) => response.json())
224 .catch((error) => console.error(error));
225 }
226
227 //Favoris
228
229 //Récupérer les favoris de l'utilisateur
230 export function getFav(idUser){
231     //Lien
232     var link = URL_WS + 'getFav.php';
233     //FormData
234     var formData = new FormData();
235     //Ajouter les données
236     formData.append('idUser', idUser);
237     //Appel fetch
238     return fetch(link, {
239         method: 'POST',
240         headers: {
241             Accept: 'application/json', //type de retour
242             'Content-Type': 'multipart/form-data' //type d'envoi
243         },
244         body: formData, //Requete
245     })
246     .then((response) => response.json())
247     .catch((error) => console.error(error));
248 }
249
250 //Checker si le favoris existe
251 export function checkFav(fav){
252     //Lien
253     var link = URL_WS + 'checkFav.php';
254     //FormData
255     var formData = new FormData();
256     //Ajouter les données
257     formData.append('idUser', fav.idUser);
258     formData.append('idImage', fav.idImage);
259     //Appel fetch
260     return fetch(link, {
261         method: 'POST',
262         headers: {
263             Accept: 'application/json', //type de retour
264             'Content-Type': 'multipart/form-data' //type d'envoi
265         },
266         body: formData, //Requete
267     })
268     .then((response) => response.json())
269     .catch((error) => console.error(error));
270 }
271
```

```
272 //loggle le favoris
273 export function toggleFav(fav){
274     //Lien
275     var link = URL_WS + 'toggleFav.php';
276     //FormData
277     var formData = new FormData();
278     //Ajouter les données
279     formData.append('idUser', fav.idUser);
280     formData.append('idImage', fav.idImage);
281     //Appel fetch
282     return fetch(link, {
283         method: 'POST',
284         headers: {
285             Accept: 'application/json', //type de retour
286             'Content-Type': 'multipart/form-data' //type d'envoi
287         },
288         body: formData, //Requete
289     })
290     .then((response) => response.json())
291     .catch((error) => console.error(error));
292 }
```

```
1 //Librairies
2 import React, { Component } from 'react';
3 import { View } from 'react-native';
4 //Components perso
5 import { BASE_STYLE, Loading, ImageList } from './MyComponent.js';
6 //Fonctions
7 import { getLastWP } from '../WS/functions.js';
8 //Redux
9 import { connect } from 'react-redux';
10
11 class Home extends Component {
12   constructor(props) {
13     super(props);
14     //State
15     this.state = {
16       wallpapers: undefined,
17       isLoading: true,
18       refreshing: false
19     };
20     //Récupérer les images
21     getLastWP().then(data => this.setState({
22       wallpapers: data.data,
23       isLoading: false
24     }));
25   }
26
27   //Quand la liste est rafraîchit
28   _onRefresh = () => {
29     this.setState({refreshing: true});
30     //Récupérer les images
31     getLastWP().then(data => this.setState({
32       wallpapers: data.data,
33       refreshing: false
34     }));
35   }
36
37   render(){
38     return (
39       <View style={BASE_STYLE.container}>
40         {/* Affichage des fonds d'écrans */}
41         <ImageList
42           data={this.state.wallpapers}
43           onPress={(item) =>
44             this.props.navigation.navigate('WallpaperDetails', {wallpaper: item})}
45           refreshing={this.state.refreshing}
46           onRefresh={() => this._onRefresh()}
47         />
48         {/* Affichage du chargement */}
49         {this.state.isLoading && <Loading/>}
50       </View>
51     );
52   }
53 }
54 //Pour Redux
```

```
55 const mapStateToProps = (state) => {  
56   return {  
57     idUser: state.isConnected.idUser  
58   }  
59 }  
60  
61 //Export du component  
62 export default connect(mapStateToProps)(Home);
```

```
1 //Librairies
2 import React from 'react';
3 //Components perso
4 import NavLogin from '../Navigation/NavLogin.js';
5 import NavLogged from '../Navigation/NavLogged.js';
6 //Redux
7 import { connect } from 'react-redux';
8
9 class Index extends React.Component {
10   render() {
11     return (
12       (this.props.idUser === false ? <NavLogin/> : <NavLogged/>)
13     )
14   }
15 }
16
17 //Pour Redux
18 const mapStateToProps = (state) => {
19   return {
20     idUser: state.isConnected.idUser
21   }
22 }
23
24 //Export du component
25 export default connect(mapStateToProps)(Index);
26
```

```
1 //Librairies
2 import React, { Component } from 'react';
3 import { View, Button, Alert } from 'react-native';
4 import { Input } from 'react-native-elements';
5 import { KeyboardAwareScrollView } from 'react-native-keyboard-aware-scroll-
view';
6 //Components perso
7 import { BASE_STYLE, Loading } from './MyComponent.js';
8 //Fonctions
9 import { login, trimValues, nothingEmpty, TEXT_FILL_ALL } from
'../WS/functions.js';
10 //Redux
11 import { connect } from 'react-redux';
12
13 class Login extends Component {
14   constructor(props) {
15     super(props);
16     //Variables
17     this.user = {
18       pseudo: '',
19       pwd: '',
20     }
21     //State
22     this.state = {
23       isLoading: false
24     };
25   }
26
27   //Login
28   _login(){
29     //Trim
30     this.user = trimValues(this.user);
31     //Aucun champs vide
32     if(nothingEmpty(this.user)){
33       //Lancer le chargement
34       this.setState({
35         isLoading: true
36       })
37       //Login
38       login(this.user).then((data) => {
39         //Fin du chargement
40         this.setState({
41           isLoading: false
42         });
43         //Si la connexion est valide
44         if(data !== false){
45           const action = { type: "IS_CONNECTED", value: data };
46           this.props.dispatch(action);
47         }else{ //Connexion invalide
48           //Envoie un message d'alerte
49           Alert.alert(
50             "Information",
51             "Invalid connection !",
52             [{text:"Ok"}]
53           );
54         }
55       });
56     }
57   }
58 }
```

```

54      },
55    });
56  });
57  }else{
58    //Envoie un message d'alerte
59    Alert.alert(
60      "Information",
61      TEXT_FILL_ALL,
62      [{text:"Ok"}]
63    );
64  }
65
66  }
67
68  render(){
69    return (
70      <View style={BASE_STYLE.container}>
71        <KeyboardAwareScrollView
72          style={BASE_STYLE.scrollview_container}
73          contentContainerStyle={BASE_STYLE.scrollview_content}
74        >
75          { /* Pseudo */ }
76          <Input
77            placeholder='Pseudo'
78            onChangeText={(value) => this.user.pseudo = value}
79          />
80          { /* Mot de passe */ }
81          <Input
82            secureTextEntry={true}
83            placeholder='Password'
84            onChangeText={(value) => this.user.pwd = value}
85          />
86          <Button
87            title='Login'
88            onPress={() => this._login()}
89          />
90          <Button
91            title='Sign up'
92            onPress={() => this.props.navigation.navigate('SignUp')}
93          />
94        </KeyboardAwareScrollView>
95        { /* Affichage du chargement */ }
96        {this.state.isLoading && <Loading/>}
97      </View>
98    );
99  }
100 }
101
102 //Pour Redux
103 const mapStateToProps = (state) => {
104   return {
105     idUser: state.isConnected.idUser
106   }
107 }
108

```



```
108  
109 //Export du component  
110 export default connect(mapStateToProps)(Login);
```

```
1 //Librairies
2 import React, { Component } from 'react';
3 import { View, Button, Alert } from 'react-native';
4 //Components perso
5 import { BASE_STYLE } from './MyComponent.js';
6 //Redux
7 import { connect } from 'react-redux';
8
9 class Logout extends Component {
10   constructor(props) {
11     super(props);
12     //State
13     this.state = {
14       isLoading: false
15     };
16   }
17
18   //Avertir avant de déconnecter
19   _logout(){
20     Alert.alert(
21       'Warning',
22       'Are you sure you want to logout ?',
23       [
24         {
25           text: 'Cancel',
26           style: 'cancel'
27         },
28         {
29           text: 'Logout',
30           onPress: () => {
31             const action = { type: "IS_CONNECTED", value: false };
32             this.props.dispatch(action);
33           }
34         }
35       ]
36     );
37   }
38
39   render(){
40     return (
41       <View style={BASE_STYLE.container}>
42         <Button
43           title='Logout'
44           onPress={() => this._logout()}
45         />
46       </View>
47     );
48   }
49 }
50
51 //Pour Redux
52 const mapStateToProps = (state) => {
53   return {
54     idUser: state.isConnected.idUser
55   }
56 }
```

```
56 }  
57  
58 //Export du component  
59 export default connect(mapStateToProps)(Logout);
```

```
1 //Librairies
2 import React from 'react';
3 import { StyleSheet, ActivityIndicator, View, FlatList, TouchableOpacity,
4   Image, RefreshControl } from 'react-native';
5 //Constantes
6 import { WIDTH_SCREEN, HEIGHT_SCREEN } from '../WS/functions.js';
7 //Style utilisé sur plusieurs pages
8 export const BASE_STYLE = StyleSheet.create({
9   //Container
10  container: {
11    flex: 1,
12    backgroundColor: '#fff'
13  },
14  center_absolute: {
15    flex: 1,
16    position: 'absolute',
17    alignItems: 'center',
18    justifyContent: 'center',
19    top: 0,
20    left: 0,
21    right: 0,
22    bottom: 0,
23  },
24  scrollview_container: {
25    flex: 1,
26  },
27  scrollview_content: {
28    alignItems: 'center',
29    margin: 15
30  },
31  //Images
32  img_left: {
33    width: (WIDTH_SCREEN / 2) - (WIDTH_SCREEN / 100),
34    height: (HEIGHT_SCREEN / 2) - (HEIGHT_SCREEN / 100),
35    marginRight: (WIDTH_SCREEN / 100),
36    marginBottom: (HEIGHT_SCREEN / 100)
37  },
38  img_right: {
39    width: (WIDTH_SCREEN / 2) - (WIDTH_SCREEN / 100),
40    height: (HEIGHT_SCREEN / 2) - (HEIGHT_SCREEN / 100),
41    marginLeft: (WIDTH_SCREEN / 100),
42    marginBottom: (HEIGHT_SCREEN / 100)
43  },
44  img_full: {
45    width: WIDTH_SCREEN,
46    height: HEIGHT_SCREEN,
47  },
48  //Informations sur le wallpaper
49  infos_wallpaper: {
50    margin: 15
51  },
52  //Textes
53  text_link: {
54    fontWeight: 'bold'.
```

```

55     textDecorationLine: 'underline'
56   },
57   text_subtitle: {
58     margin: 15,
59     fontSize: 20,
60     fontWeight: 'bold'
61   }
62 });
63
64 //Components
65
66 //Affichage d'une icône de chargement
67 export function Loading(){
68   return(
69     <View style={BASE_STYLE.center_absolute}>
70       <ActivityIndicator size='large' />
71     </View>
72   );
73 }
74
75 //Affichage de plusieurs images en liste
76 export function ImageList(props){
77   //Si les données ne sont pas undefined
78   if(props.data !== undefined){
79     return(
80       <FlatList
81         style={{flex: 1}}
82         data={props.data}
83         refreshControl={
84           <RefreshControl
85             refreshing={props.refreshing}
86             onRefresh={props.onRefresh}
87           />
88         }
89         renderItem={({item, index}) => (
90           <TouchableOpacity
91             onPress={() => props.onPress(item)}
92             >
93             <Image
94               style={index % 2 == 0 ? BASE_STYLE.img_left :
BASE_STYLE.img_right}
95               source={{uri: item.thumbs.original}}
96             />
97             </TouchableOpacity>
98           )}
99         numColumns={2}
100       />
101     );
102   }else{
103     return null;
104   }
105 }

```

```
1 //Librairies
2 import React from 'react';
3 import { Entypo, MaterialCommunityIcons, Ionicons } from 'react-native-
vector-icons';
4 //Components perso
5 import { HomeStack } from './SharedNav.js';
6 import Fav from '../Components/Fav.js';
7 import Search from '../Components/Search.js';
8 import SearchResult from '../Components/SearchResult.js';
9 import WallpaperDetails from '../Components/WallpaperDetails.js';
10 import Logout from '../Components/Logout.js';
11 //Constantes
12 import { COLOR_FOCUSED, COLOR_NOT_FOCUSED, Stack, Tab } from
"../WS/functions.js";
13
14 //Favoris
15 function FavStack(){
16     return(
17         <Stack.Navigator>
18             <Stack.Screen name="Favorites" component={Fav} />
19             <Stack.Screen
20                 name="WallpaperDetails"
21                 options={{title: 'Details', headerBackTitle: 'Back'}}
22                 component={WallpaperDetails} />
23         </Stack.Navigator>
24     );
25 }
26
27 //Recherche
28 function SearchStack(){
29     return(
30         <Stack.Navigator>
31             <Stack.Screen name="Search" component={Search} />
32             <Stack.Screen
33                 name="SearchResult"
34                 options={{title: 'Search result', headerBackTitle: 'Back'}}
35                 component={SearchResult} />
36             <Stack.Screen
37                 name="WallpaperDetails"
38                 options={{title: 'Details', headerBackTitle: 'Back'}}
39                 component={WallpaperDetails} />
40         </Stack.Navigator>
41     );
42 }
43
44 //Logout
45 function LogoutStack(){
46     return(
47         <Stack.Navigator>
48             <Stack.Screen name="Logout" component={Logout} />
49         </Stack.Navigator>
50     );
51 }
52
53 //Navigation
```

```
54 function NavLogged(){
55     return(
56         <Tab.Navigator>
57             {/* Page d'accueil */}
58             <Tab.Screen
59                 name="Home"
60                 component={HomeStack}
61                 options={{
62                     tabBarIcon: ({focused}) => {
63                         return(
64                             <Entypo
65                                 name="home"
66                                 size={25}
67                                 color={focused ? COLOR_FOCUSED : COLOR_NOT_FOCUSED}
68                             />
69                         )
70                     }
71                 }}
72             />
73             {/* Favoris */}
74             <Tab.Screen
75                 name="Favorites"
76                 component={FavStack}
77                 options={{
78                     tabBarIcon: ({focused}) => {
79                         return(
80                             <Entypo
81                                 name="star"
82                                 size={25}
83                                 color={focused ? COLOR_FOCUSED : COLOR_NOT_FOCUSED}
84                             />
85                         )
86                     }
87                 }}
88             />
89             {/* Recherche */}
90             <Tab.Screen
91                 name="Search"
92                 component={SearchStack}
93                 options={{
94                     tabBarIcon: ({focused}) => {
95                         return(
96                             <Icons
97                                 name="ios-search"
98                                 size={25}
99                                 color={focused ? COLOR_FOCUSED : COLOR_NOT_FOCUSED}
100                             />
101                         )
102                     }
103                 }}
104             />
105             {/* Logout */}
106             <Tab.Screen
107                 name="Logout"
108                 component={LogoutStack}
109             />
110         </Tab.Navigator>
111     )
112 }
```

```
108     component={LogoutStack}
109     options={{
110       tabBarIcon: ({focused}) => {
111         return(
112           <MaterialCommunityIcons
113             name="logout"
114             size={25}
115             color={focused ? COLOR_FOCUSED : COLOR_NOT_FOCUSED}
116           />
117         )
118       }
119     }}
120   />
121 </Tab.Navigator>
122 );
123 }
124
125 //Export du component
126 export default NavLogged;
```



```
1 //Librairies
2 import React from 'react';
3 import { Entypo } from 'react-native-vector-icons';
4 //Components perso
5 import { HomeStack } from './SharedNav.js';
6 import Login from '../Components/Login.js';
7 import SignUp from '../Components/SignUp.js';
8 //Constantes
9 import { COLOR_FOCUSED, COLOR_NOT_FOCUSED, Stack, Tab } from
  "../WS/functions.js";
10
11 //Login
12 function LoginStack(){
13   return(
14     <Stack.Navigator>
15       <Stack.Screen name="Login" component={Login} />
16       <Stack.Screen
17         name="SignUp"
18         options={{title: 'Sign Up'}}
19         component={SignUp} />
20     </Stack.Navigator>
21   );
22 }
23
24 //Navigation
25 function NavLogin(){
26   return(
27     <Tab.Navigator>
28       <Tab.Screen
29         name="Home"
30         component={HomeStack}
31         options={{
32           tabBarIcon: ({focused}) => {
33             return(
34               <Entypo
35                 name="home"
36                 size={25}
37                 color={focused ? COLOR_FOCUSED : COLOR_NOT_FOCUSED}
38               />
39             )
40           }
41         }}
42       />
43       <Tab.Screen
44         name="Login"
45         component={LoginStack}
46         options={{
47           tabBarIcon: ({focused}) => {
48             return(
49               <Entypo
50                 name="login"
51                 size={25}
52                 color={focused ? COLOR_FOCUSED : COLOR_NOT_FOCUSED}
53               />
54             )
55           }
56         }}
57       />
58     </Tab.Navigator>
59   );
60 }
```

```
55         }  
56     }}  
57     />  
58     </Tab.Navigator>  
59 );  
60 }  
61  
62 //Export du component  
63 export default NavLogin;
```

```

1 //Librairies
2 import React, { Component } from 'react';
3 import { Text, View, Button, TouchableOpacity } from 'react-native';
4 import { Input } from 'react-native-elements';
5 import { KeyboardAwareScrollView } from 'react-native-keyboard-aware-scroll-
view';
6 //Components perso
7 import { BASE_STYLE, Loading } from './MyComponent.js';
8 //Fonctions
9 import { CATEGORIES, ORDER_DESC, ORDER_ASC, SORTING } from
'../WS/functions.js';
10 //Redux
11 import { connect } from 'react-redux';
12
13 class Search extends Component {
14   constructor(props) {
15     super(props);
16     //Recherche
17     this.search = {
18       tag: null,
19       owner: null,
20       color: null,
21     }
22     //State
23     this.state = {
24       isLoading: false,
25       category: CATEGORIES,
26       order: ORDER_DESC,
27       sorting: null
28     };
29   }
30
31   //Effectuer la recherche
32   _search(){
33     //Récupérer les valeurs des catégories
34     var categories = '';
35     this.state.category.map(categorie => categories += categorie.value);
36
37     //Page suivante
38     this.props.navigation.navigate('SearchResult', {search:
39     {...this.search, category: categories, order: this.state.order, sorting:
40     this.state.sorting}})
41   }
42
43   render(){
44     return (
45       <View style={BASE_STYLE.container}>
46         <KeyboardAwareScrollView
47         style={BASE_STYLE.scrollview_container}
48         contentContainerStyle={BASE_STYLE.scrollview_content}
49         >
50           { /* Catégories de recherches */}
51           <Text style={BASE_STYLE.text_subtitle}>Categories</Text>
52           <View style={{flexDirection: 'row'}}>
53             {this.state.category.map(categorie => (

```

```

52         <TouchableOpacity
53         key={categorie.name}
54         style={{flex: 1}}
55         onPress={() => {
56             //Copier les catégories dans la variable de
résultat
57             var res = [...this.state.category];
58             //Trouver l'élément cliqué
59             var indexElem = res.findIndex(elem =>
elem.name == categorie.name);
60             //Inverser sa valeur
61             res[indexElem].value = (res[indexElem].value
== 0 ? 1 : 0);
62
63             //Mettre à jour la valeur
64             this.setState({
65                 category: res
66             });
67         }}
68         >
69         <Text style={{textAlign: 'center',
fontWeight: (categorie.value == 1 ? 'bold' : 'normal')}}>{categorie.name}
</Text>
70     </TouchableOpacity>
71     )))
72 </View>
73 {/ * Sorting */}
74 <Text style={BASE_STYLE.text_subtitle}>Sorting</Text>
75 {SORTING.map(typeSorting => (
76     <TouchableOpacity
77     key={typeSorting}
78     style={{marginVertical: 5}}
79     onPress={() => this.setState({sorting: typeSorting})}
80     >
81     <Text style={{textAlign: 'center', fontWeight:
(typeSorting == this.state.sorting ? 'bold' : 'normal')}}>{typeSorting}
</Text>
82     </TouchableOpacity>
83     )))
84 {/ * Order */}
85 <Text style={BASE_STYLE.text_subtitle}>Order</Text>
86 <TouchableOpacity
87     onPress={() => this.setState({
88         order: (this.state.order == ORDER_DESC ? ORDER_ASC :
ORDER_DESC)
89     })}
90     >
91     <Text>{this.state.order}</Text>
92 </TouchableOpacity>
93
94 {/ * Tag */}
95 <Input
96     placeholder='Tag'
97     onChangeText={(value) => this.search.tag = value}
98     autoCapitalize='none'

```

```
98         autoCapitalize='none'
99         autoCorrect={false}
100     />
101     <Text>Which tag are you looking for ?</Text>
102     { /* Owner */ }
103     <Input
104     placeholder='Owner'
105     onChangeText={(value) => this.search.owner = value}
106     autoCapitalize='none'
107     autoCorrect={false}
108     />
109     <Text>Owner of the wallpaper</Text>
110     { /* Color */ }
111     <Input
112     placeholder='Color'
113     onChangeText={(value) => this.search.color = value}
114     autoCapitalize='none'
115     autoCorrect={false}
116     />
117     <Text>Hexa value of the color without the #</Text>
118     { /* Recherche */ }
119     <Button
120     title='Search'
121     onPress={() => this._search()}
122     />
123     </KeyboardAwareScrollView>
124     { /* Affichage du chargement */ }
125     {this.state.isLoading && <Loading/>}
126 </View>
127 );
128 }
129 }
130
131 //Pour Redux
132 const mapStateToProps = (state) => {
133     return {
134         idUser: state.isConnected.idUser
135     }
136 }
137
138 //Export du component
139 export default connect(mapStateToProps)(Search);
```

```
1 //Librairies
2 import React, { Component } from 'react';
3 import { Text, View, TouchableOpacity } from 'react-native';
4 import { Entypo } from 'react-native-vector-icons';
5 //Components perso
6 import { BASE_STYLE, Loading, ImageList } from './MyComponent.js';
7 //Fonctions
8 import { searchWP } from '../WS/functions.js';
9 //Redux
10 import { connect } from 'react-redux';
11
12 const SIZE_CHEVRON = 30;
13
14 class SearchResult extends Component {
15   constructor(props) {
16     super(props);
17     //State
18     this.state = {
19       wallpapers: undefined,
20       isLoading: true,
21       refreshing: false,
22       currentPage: 0,
23       lastPage: 0,
24     };
25     //Récupérer les images
26     searchWP(this.props.route.params.search).then(data => this.setState({
27       wallpapers: data.data,
28       currentPage: data.meta.current_page,
29       lastPage: data.meta.last_page,
30       isLoading: false
31     }));
32   }
33
34   //Quand la liste est rafraîchit
35   _onRefresh = () => {
36     this.setState({refreshing: true});
37     //Récupérer les images
38     searchWP(this.props.route.params.search,
39 this.state.currentPage).then(data => this.setState({
40       wallpapers: data.data,
41       currentPage: data.meta.current_page,
42       lastPage: data.meta.last_page,
43       refreshing: false
44     }));
45   }
46
47   //Changer de page
48   _changePage(page){
49     //Si la page demandé reste dans les limites
50     if(page > 0 && page <= this.state.lastPage){
51       this.setState({isLoading: true});
52       //Récupérer les images
53       searchWP(this.props.route.params.search, page).then(data =>
54 this.setState({
55       wallpapers: data.data
```

```

53         wallpaper: wallpaper,
54         currentPage: data.meta.current_page,
55         lastPage: data.meta.last_page,
56         isLoading: false
57     }));
58 }
59 }
60
61 render(){
62     return (
63         <View style={BASE_STYLE.container}>
64             {/* Affichage des résultats, ou d'un message indiquant qu'il
n'y a pas de résultat */}
65             {(this.state.wallpapers != undefined &&
this.state.wallpapers.length == 0) ?
66                 <View style={{flex: 1, justifyContent: 'center',
alignItems: 'center'}}>
67                     <Text>There's nothing here...</Text>
68                 </View>
69                 :
70                 <ImageList
71                     data={this.state.wallpapers}
72                     onPress={(item) =>
this.props.navigation.navigate('WallpaperDetails', {wallpaper: item})}
73                     refreshing={this.state.refreshing}
74                     onRefresh={() => this._onRefresh()}
75                 />
76             }
77             {/* Pagination */}
78             <View style={{flexDirection: 'row', justifyContent: 'space-
around', alignItems: 'center'}}>
79                 {/* Page précédente */}
80                 <TouchableOpacity
81                 onPress={() => this._changePage(this.state.currentPage -
1)}>
82                     >
83                     <Entypo
84                         name="chevron-thin-left"
85                         size={SIZE_CHEVRON}
86                     />
87                 </TouchableOpacity>
88                 {/* Page actuelle */}
89                 <Text>
90                     Current page:
91                     {this.state.currentPage}/{this.state.lastPage}
92                 </Text>
93                 {/* Page suivante */}
94                 <TouchableOpacity
95                 onPress={() => this._changePage(this.state.currentPage +
1)}>
96                     >
97                     <Entypo
98                         name="chevron-thin-right"
99                         size={SIZE_CHEVRON}
100                     />
101                 </TouchableOpacity>

```

```
100         </TouchableOpacity>
101     </View>
102     { /* Affichage du chargement */
103     {this.state.isLoading && <Loading/>}
104 </View>
105     );
106 }
107 }
108
109 //Pour Redux
110 const mapStateToProps = (state) => {
111     return {
112         idUser: state.isConnected.idUser
113     }
114 }
115
116 //Export du component
117 export default connect(mapStateToProps)(SearchResult);
```



```
1 //Librairies
2 import React from 'react';
3 //Components perso
4 import Home from '../Components/Home.js';
5 import WallpaperDetails from '../Components/WallpaperDetails.js';
6 //Constantes
7 import { Stack } from "../WS/functions.js";
8
9 //Home
10 export function HomeStack(){
11     return(
12         <Stack.Navigator>
13             <Stack.Screen name="Home" component={Home} />
14             <Stack.Screen
15                 name="WallpaperDetails"
16                 options={{title: 'Details', headerBackTitle: 'Back'}}
17                 component={WallpaperDetails} />
18         </Stack.Navigator>
19     );
20 }
```

```
1 //Librairies
2 import React, { Component } from 'react';
3 import { View, Button, Alert } from 'react-native';
4 import { Input } from 'react-native-elements';
5 import { KeyboardAwareScrollView } from 'react-native-keyboard-aware-scroll-
view';
6 //Components perso
7 import { BASE_STYLE, Loading } from './MyComponent.js';
8 //Fonctions
9 import { signUp, trimValues, nothingEmpty, TEXT_FILL_ALL } from
'../WS/functions.js';
10 //Redux
11 import { connect } from 'react-redux';
12
13 class SignUp extends Component {
14   constructor(props) {
15     super(props);
16     //Variables
17     this.user = {
18       pseudo: '',
19       email: '',
20       pwd: '',
21     }
22     //State
23     this.state = {
24       isLoading: false
25     };
26   }
27
28   //Inscription
29   _signUp(){
30     //Trim
31     this.user = trimValues(this.user);
32     //Aucun champs vide
33     if(nothingEmpty(this.user)){
34       //Lancer le chargement
35       this.setState({
36         isLoading: true
37       });
38       //Inscription
39       signUp(this.user).then((data) => {
40         //Fin du chargement
41         this.setState({
42           isLoading: false
43         });
44         //Si l'inscription est valide
45         if(data.done === true){
46           //Envoie un message d'alerte
47           Alert.alert(
48             "Information",
49             "Successful sign up",
50             [{
51               text: "Ok",
52               onPress: () => {
53                 const action = { type: "TS_CONNECTED", value:
```

```

53     data.msg };
54         this.props.dispatch(action);
55     }
56     }]
57     );
58     }else{ //Inscription invalide
59         //Envoie un message d'alerte
60         Alert.alert(
61             "Information",
62             data.msg,
63             [{text:"Ok"}]
64         );
65     }
66     });
67 }else{
68     //Envoie un message d'alerte
69     Alert.alert(
70         "Information",
71         TEXT_FILL_ALL,
72         [{text:"Ok"}]
73     );
74 }
75 }
76
77 render(){
78     return (
79         <View style={BASE_STYLE.container}>
80             <KeyboardAwareScrollView
81                 style={BASE_STYLE.scrollview_container}
82                 contentContainerStyle={BASE_STYLE.scrollview_content}
83             >
84                 { /* Pseudo */ }
85                 <Input
86                     placeholder='Pseudo'
87                     onChangeText={(value) => this.user.pseudo = value}
88                 />
89
90                 { /* Email */ }
91                 <Input
92                     keyboardType='email-address'
93                     placeholder='Email'
94                     onChangeText={(value) => this.user.email = value}
95                 />
96                 { /* Mot de passe */ }
97                 <Input
98                     secureTextEntry={true}
99                     placeholder='Password'
100                     onChangeText={(value) => this.user.pwd = value}
101                 />
102                 <Button
103                     title='Sign up'
104                     onPress={() => this._signUp()}
105                 />
106             </KeyboardAwareScrollView>
107             { /* Affichage du chargement */ }

```

```
107         // Affichage du chargement //
108         {this.state.isLoading && <Loading/>}
109     </View>
110     );
111 }
112 }
113
114 //Pour Redux
115 const mapStateToProps = (state) => {
116     return {
117         idUser: state.isConnected.idUser
118     }
119 }
120
121 //Export du component
122 export default connect(mapStateToProps)(SignUp);
```

```
1 //Librairies
2 import React, { Component } from 'react';
3 import { StyleSheet, Text, View, ScrollView, Button, FlatList, Image,
4 TouchableOpacity, Share } from 'react-native';
5 import * as Linking from 'expo-linking';
6 import { Entypo } from 'react-native-vector-icons';
7 //Components perso
8 import { BASE_STYLE, Loading, ImageList } from './MyComponent.js';
9 //Fonctions
10 import { getSimilarWP, checkFav, toggleFav } from '../WS/functions.js';
11 //Redux
12 import { connect } from 'react-redux';
13
14 class WallpaperDetails extends Component {
15   constructor(props) {
16     super(props);
17     //State
18     this.state = {
19       wallpaper: this.props.route.params.wallpaper,
20       similars: undefined,
21       fav: false
22     },
23     //Si l'utilisateur est connecté
24     (this.props.idUser !== false &&
25       //Checker si l'image est en favoris
26       checkFav({idUser: this.props.idUser, idImage:
27 this.state.wallpaper.id}).then(data => {
28       this.setState({fav: data});
29     })),
30     //Récupérer les images similaires
31     getSimilarWP(this.state.wallpaper.id).then(data => {
32       this.setState({similars: data.data});
33     })
34   );
35 }
36
37 //Partage du wallpaper
38 _share(){
39   Share.share({
40     message: "Wallhaven wallpaper : " + this.state.wallpaper.url,
41   });
42 }
43
44 //Toggle le favoris
45 _toggleFav(){
46   toggleFav({idUser: this.props.idUser, idImage:
47 this.state.wallpaper.id}).then(data => {
48     this.setState({fav: data});
49   });
50 }
51
52 //Ouvrir dans Wallhaven
53 _openInWallhaven(){
54   Linking.openURL(this.state.wallpaper.url);
55 }
```

```

53
54 render(){
55     return (
56         <View style={BASE_STYLE.container}>
57             <ScrollView>
58                 {/* Image */}
59                 <Image
60                     style={BASE_STYLE.img_full}
61                     source={{uri: this.state.wallpaper.path}}
62                 />
63                 {/* Informations sur le wallpaper */}
64                 <View style={BASE_STYLE.infos_wallpaper}>
65                     {/* Définition */}
66                     <Text>Resolution: {this.state.wallpaper.resolution}
67                 </Text>
68                     {/* Ouvrir dans Wallhaven */}
69                     <TouchableOpacity
70                         onPress={() => this._openInWallhaven()}
71                     >
72                         <Text style={BASE_STYLE.text_link}>Open in
73                         Wallhaven</Text>
74                     </TouchableOpacity>
75                     <View style={{flexDirection: 'row', justifyContent:
76                         'space-between'}}>
77                         {/* Partage */}
78                         <TouchableOpacity
79                             onPress={() => this._share()}
80                         >
81                             <Entypo
82                                 name="share"
83                                 size={25}
84                             />
85                         </TouchableOpacity>
86                         {/* Étoile de favoris */}
87                         {this.props.idUser !== false &&
88                             <TouchableOpacity
89                                 onPress={() => this._toggleFav()}
90                             >
91                                 <Entypo
92                                     name={this.state.fav ? "star" : "star-
93                                     outlined"}
94                                     size={25}
95                                 />
96                             </TouchableOpacity>
97                         }
98                     </View>
99                 </View>
100                 {/* Affichage des fonds d'écrans similaire seulement si
101                 l'utilisateur est connecté */}
102                 {this.props.idUser !== false &&
103                     <View>
104                         <Text style={BASE_STYLE.text_subtitle}>Similar
105                         wallpaper :</Text>
106                         <ImageList
107                             data={this.state.similars}

```

```
101         data={this.state.similaires}
102         onPress={(item) =>
this.props.navigation.push('WallpaperDetails', {wallpaper: item})}
103     />
104     </View>
105 }
106
107 </ScrollView>
108 { /* Affichage du chargement */ }
109 {this.state.isLoading && <Loading/>}
110 </View>
111 );
112 }
113 }
114
115 //Pour Redux
116 const mapStateToProps = (state) => {
117     return {
118         idUser: state.isConnected.idUser
119     }
120 }
121
122 //Export du component
123 export default connect(mapStateToProps)(WallpaperDetails);
```