**TRƯỜNG CAO ĐẲNG CÔNG NGHỆ THỦ ĐỨC**

**KHOA CÔNG NGHỆ THÔNG TIN**

**🙜🕮🙞**

****

**BÁO CÁO MÔN HỌC**

**Lập trình di động 3**

**Đề tài: Quản lí nhân sự**

**Giảng viên hướng dẫn:** Trương Bá Thái

**Nhóm sinh viên thực hiện**

* + - * Võ Cao Thanh Phong 17211TT3348
      * Đào Duy Phương 17211TT
      * Lê Thanh Liêm 17211TT

*Tp. Hồ Chí Minh*, ngày 17 tháng 10 năm 2019

**Mục lục**

[**CHƯƠNG 1: TỔNG QUAN** 3](#_Toc23345526)

[**1.** **Màn hình đăng nhập (Võ Cao Thanh Phong)** 3](#_Toc23345527)

[**2.** **Màn hình tạo tài khoản (Võ Cao Thanh Phong)** 3](#_Toc23345528)

[**3.** **Màn hình chính(Phòng ban) (Đào Duy Phương)** 4](#_Toc23345529)

[**4.** **Màn hình thêm phòng ban (Đào Duy Phương)** 5](#_Toc23345530)

[**5.** **Màn hình nhân viên trong phòng ban (Lê Thanh Liêm)** 6](#_Toc23345531)

[**6.** **Màn hình thêm nhân viên (Lê Thanh Liêm)** 8](#_Toc23345532)

[**7.** **Màn hình thông tin nhân viên (Lê Thanh Liêm)** 9](#_Toc23345533)

[**CHƯƠNG 2: ES6** 10](#_Toc23345534)

[**1.** **Biến Let, Var, Conts trong ES6 (Lê Thanh Liêm)** 10](#_Toc23345535)

[**2.** **Arrow function (Võ Cao Thanh Phong)** 10](#_Toc23345536)

[**3.** **Kiểu DL Object (Đào Duy Phương)** 10](#_Toc23345537)

[**4.** **Các bài tập es6 đã tìm hiểu:** 11](#_Toc23345538)

[**CHƯƠNG 3: COMPONENT** 12](#_Toc23345539)

[**1.** **Tổng Quan** 12](#_Toc23345540)

[**2.** **Các component cơ bản (Võ Cao Thanh Phong)** 16](#_Toc23345541)

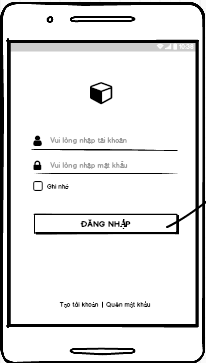
[**3.** **Các component nâng cao(Đào Duy Phương, Lê Thanh Liêm)** 18](#_Toc23345542)

# **CHƯƠNG 1: TỔNG QUAN**

***Giới thiệu đề tài***

Dân số Việt Nam ngày càng tăng cao, yêu cầu đặt ra phải giải quyết vấn đề dân số. Dân số tăng nhanh kéo theo vấn đề việc làm. Để giúp việc quản lí nhân sự của các công ty, xí nghiệp chúng tôi đã tạo ra một ứng dụng quản lí nhân sự. Ứng dụng quản lí này giúp quản lí, sắp xếp, tìm kiếm các nhân viên, công nhân của các bộ phận, các phòng ban trong công ty, xí nghiệp. Quản lí các hoạt động thông tin của nhân viên.

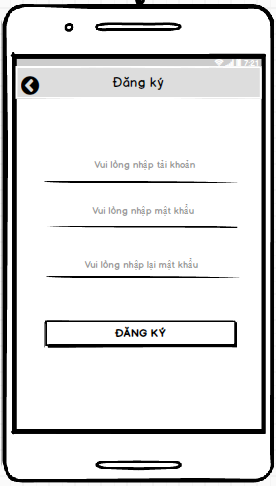
1. **Màn hình đăng nhập (Võ Cao Thanh Phong)**
   1. Giao diện



* 1. Yêu cầu chức năng

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Stt | Tên | Mô tả | Hoạt động | Trả về |
| 1 | Màn hình đăng nhập | Gồm:   * Logo app * 2 edit text(User và pass word) * Button đăng nhập * Button tạo tài khoản * Button quên mật khẩu * Checkbox | N/A | N/A |
|  | Edit text “user ” | Nhập tài khoản người dùng | Chọn và nhập | N/A |
|  | Edit text “password ” | Nhập mật khẩu người dùng | Chọn và nhập | N/A |
|  | Button đăng nhập | Kiểm tra thông tin đăng nhập | Click chọn | Thông tin đăng nhập đúng thì đăng nhập thành công ngược lại thông báo lỗi sai tài khoản hoặc mật khẩu |
|  | Checkbox ghi nhớ | Lưu tài khoản | Chọn |  |

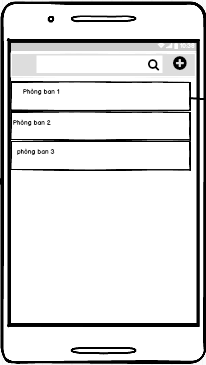
1. **Màn hình tạo tài khoản (Võ Cao Thanh Phong)**

a. Giao diện

b.Yêu cầu chức năng

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Stt | Tên | Mô tả | Hoạt động | Trả về |
| 1 | Màn hình đăng nhập | Gồm:   * 3 edit text(nhập tài khoản,nhập mật khẩu,xác nhận lại mật khẩu) * Button đăng ký | N/A | N/A |
|  | Edit text “tài khoản ” | Nhập tài khoản người dùng | Chọn và nhập | N/A |
|  | Edit text “mật khẩu ” | Nhập mật khẩu người dùng | Chọn và nhập | N/A |
|  | Edit text “xác nhận lại mật khẩu” | Nhập lại mật khẩu | Chọn và nhập | N/A |
|  | Button đăng ký | Tạo tài khoản mới | Chọn button | N/A |

1. **Màn hình chính(Phòng ban) (Đào Duy Phương)**
   1. Giao diện



* 1. Yêu cầu chức năng

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Stt | Tên | Mô tả | Hoạt động | Trả về |
| 1 | Màn hình phòng ban | Gồm:   * List view phòng ban * Thanh tìm kiếm * Button thêm phòng ban | N/A | N/A |
| 2 | Listview | Hiện danh sách các phòng ban | Nhấn vào các item có trong listview | Hiển thị Listview nhân viên trong phòng ban đó |
| 3 | Thanh tìm kiếm | Tìm kiếm tên phòng ban | Tìm kiếm theo tên phòng | Trả về tên phòng |
| 4 | Button thêm phòng ban | Thêm dữ liệu phòng ban | Thêm dữ liệu | Tạo thêm item trong listview rồi trả về giá trị |

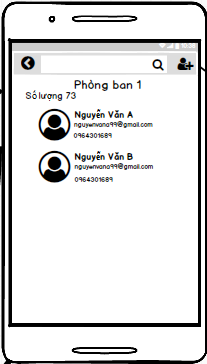
1. **Màn hình thêm phòng ban (Đào Duy Phương)**
2. Giao diện



1. Yêu cầu chức năng

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Stt | Tên | Mô tả | Hoạt động | Trả về |
| 1 | Màn hình thêm phòng ban | Gồm:   * 2 edit text * Text view số lượng nhân viên * Button thêm nhân viên * Button trở về | N/A | N/A |
| 2 | Button trở về | Trở về màn hình trước | Trở về | N/A |
| 3 | Edit Mã phòng | Thêm mã phòng | Thêm mã phòng | N/A |
| 4 | Thanh tìm kiếm | Thêm tên phòng | Thêm tên phòng | N/A |
| 5 | Button Thêm nhân viên | Thêm phòng ban | Thêm phòng ban | Thêm một phòng mới vào dữ liệu |

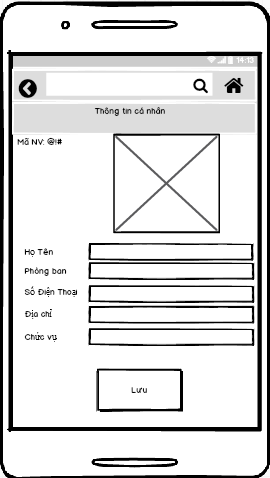
1. **Màn hình nhân viên trong phòng ban (Lê Thanh Liêm)**
2. Giao diện



1. Yêu cầu chức năng

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Stt | Tên | Mô tả | Hoạt động | Trả về |
| 1 | Màn hình nhân viên trong phòng ban | Gồm:   * List view hiên nhân viên trong phòng ban tương ứng * Thanh tìm kiếm * Text view tên phòng ban * Text view số lượng nhân viên * Button thêm nhân viên * Button trở về | N/A | N/A |
| 2 | Button trở về | Trở về màn hình trước | Trở về | N/A |
| 3 | Item danh sách nhân viên | Hiện nhân viên | Hiện lên danh sách nhân viên | Khi nhấn vào sẽ hiện lên chi tiết nhân viên |
| 4 | Thanh tìm kiếm | Tìm kiếm nhân viên | Tìm kiếm theo tên | Trả về nhân viên có từ khóa gần hoặc đúng nhất |
| 5 | Button Thêm nhân viên | Thêm nhân viên | Thêm nhân viên | Thêm một nhân viên mới vào dữ liệu |
| 6 | Số lượng nhân viên | Cho biết số lượng nhân viên trong phòng ban | Hiện số lựợng | Hiện số lượng được đếm trong listview |

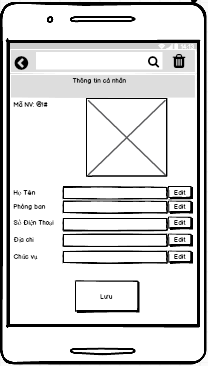
1. **Màn hình thêm nhân viên (Lê Thanh Liêm)**
2. Giao diện



1. Yêu cầu chức năng

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Stt | Tên | Mô tả | Hoạt động | Trả về |
| 1 | Màn hình thêm nhân viên | Gồm:   * Các Text view * Có Image(Hình ảnh nhân viên) * Các Edit text * Button lưu | N/A | N/A |
| 2 | Các text view | Hiển thị các mục thông tin nhân viên | N/A | N/A |
| 3 | Image View | Thêm hình ảnh nhân viên | Click chọn | Hiện màn hình thêm hình |
| 4 | Button lưu | Lưu thông tin nhân viên vào danh sách | Click chọn | Màn hình nhân viên của phòng ban đó |

1. **Màn hình thông tin nhân viên (Lê Thanh Liêm)**
2. Giao diện



1. Yêu cầu chức năng

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Stt | Tên | Mô tả | Hoạt động | Trả về |
| 1 | Màn hình thông tin nhân viên | Gồm:   * Các Text view * Có Image(Hình ảnh nhân viên) * Các Edit text * Button lưu thêm phòng ban * Button xóa | N/A | N/A |
| 2 | Listview | Hiển thị các mục thông tin nhân viên | N/A | N/A |
| 3 | Thanh tìm kiếm | Thêm hình ảnh nhân viên | Click chọn | Hiện màn hình thêm hình |
| 4 | Button lưu | Lưu thông tin nhân viên vào danh sách | Click chọn | Màn hình nhân viên của phòng ban đó |
| 5 | Button xóa | Xóa nhân viên | Click chọn | Nhân viên bị xóa khỏi dữ liệu |

# **CHƯƠNG 2: ES6**

1. ***Biến Let, Var, Conts trong ES6 (Lê Thanh Liêm)***

* Biến let: từ khóa let dùng để khởi tạo một biến nhưng biến đó chỉ có tác dụng bên trong khối đang khai báo

if (a < b)

{

    let tmp = a;

    //biến let chỉ tồn tại trong phạm vi này

}

* Biến var: Dùng chung trong toàn bộ chương trình

var a = 12;

var b = 20;

* Conts: Hằng

1. ***Arrow function (Võ Cao Thanh Phong)***

Arrow function - còn được gọi là "fat arrow", là cú pháp được mượn từ CoffeeScript (một ngôn ngữ chuyển tiếp), cú pháp này là cách ngắn gọn hơn dùng để viết function. Ở đây sử dụng kí tự =>, trông giống như một mũi tên "béo". Arrow function là một hàm vô danh và nó thay đổi cách this bind đến function. Arrow function làm code của ta trông ngắn gọn hơn, giúp đơn giản hóa function scoping cũng như từ khóa this. Arrow function hoạt động tương tự như Lambdas trong các ngôn ngữ khác như C # hay Python. Bằng cách sử dụng arrow function, chúng ta tránh được việc phải gõ từ khoá function, return và dấu ngoặc nhọn.

* Cú pháp căn bản**:**

var functionName = (var1, var2) => {

    // Nội dung function

};

* Ví dụ:

var hello = (name, message) => {

    console.log("Chào " + name + ", bạn là " + message);

};

hello('Cường', 'admin freetuts.net');

1. ***Kiểu DL Object (Đào Duy Phương)***

Đối tượng là một khái niệm trừu tượng thể hiện cho một đối tượng cụ thể mà chúng ta có thể tự tạo một đối tượng theo ý của mình dựa vào yêu cầu của ứng dụng.

**Cách 1**: Sử dụng từ khóa new Object().

var Students= new Object();

**` Cách 2**: Sử dụng từ khóa {}.

var Students={};

1. ***Các bài tập es6 đã tìm hiểu:***

* ***Võ Cao Thanh Phong:***

var a = 10

var b = -15

var pt = 3

console.log("Số a:", a, "; Số b:", b)

var TinhTong = (SoA, SoB, pt) => {

switch (pt){

case 1:

console.log("Tổng a + b là:", a + b);

break;

case 2:

console.log("Hiệu a - b là:", a - b);

break;

case 3:

console.log("Tích a \* b là:", a \* b);

break

case 4:

console.log("Thương a / b là:", a \* b);

break

}

};

TinhTong(a,b, pt);

* ***Đào Duy Phương***

var numbers = new Set([1, 2, 3, 4]);

// them phần tử

numbers.add(5)

//xóa phan tu

numbers.delete(2);

//kiem tra phan tu có ton tai hay chua neu có tra gia tri true ngươc lại tra flase

console.log(numbers.has(1));

* ***Lê Thanh Liêm***

var a;

var b = 6, c = 80;

var ptCong = (b, c)=>{

return a + b;

};

var ptTru = (b, c)=>{

return b - c;

};

var ptNhan = (b, c)=>{

return b \* c;

};

var ptChia = (b, c)=>{

return b / c;

};

switch(3)

{

case 1:

ptCong(b,c);

break;

case 2:

ptTru(b,c);

break;

case 3:

ptNhan(b,c);

break;

case 4:

ptChia(b,c);

break;

# **CHƯƠNG 3: COMPONENT**

1. **Tổng Quan**

Trong ReactJS, mỗi đoạn code sẽ được phân chia thành những Component không lệ thuộc lẫn nhau và có thể tái sử dụng khi cần thiết.

***Props(Võ Cao Thanh Phong)***

Các component nhận props từ component cha. Chúng ta không được thay đổi giá trị của props trong các component này mà chỉ được phép đọc giá trị ra thôi. Trong React thì dữ liệu sẽ đi theo một chiều, có nghĩa là từ component cha => các component con.

Ví dụ:

import React, { Component } from 'react';

import { Text, View } from 'react-native';

class Greeting extends Component {

render() {

return (

<View style={{alignItems: 'center'}}>

<Text>Hello {this.props.name}!</Text>

</View>

);

}

}

export default class LotsOfGreetings extends Component {

render() {

return (

<View style={{alignItems: 'center', top: 50}}>

<Greeting name='Rexxar' />

<Greeting name='Jaina' />

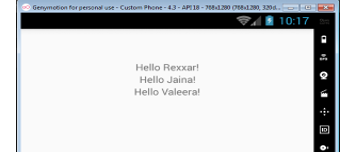
<Greeting name='Valeera' />

</View>

);

}

}



***State(Võ Cao Thanh Phong)***

State thì hoạt động khác với props. state là dữ liệu nội bộ của một Component, trong khi props là dữ liệu được truyền cho Component. Chính vì vậy chúng ta hoàn toàn có thể thay đổi state

Ví dụ:

import React, { Component } from 'react';

import { Text, View } from 'react-native';

class Blink extends Component {

componentDidMount(){

// Toggle the state every second

setInterval(() => (

this.setState(previousState => (

{ isShowingText: !previousState.isShowingText }

))

), 1000);

}

//state object

state = { isShowingText: true };

render() {

if (!this.state.isShowingText) {

return null;

}

return (

<Text>{this.props.text}</Text>

);

}

}

export default class BlinkApp extends Component {

render() {

return (

<View>

<Blink text='I love to blink' />

<Blink text='Yes blinking is so great' />

<Blink text='Why did they ever take this out of HTML' />

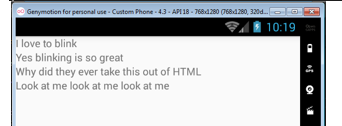
<Blink text='Look at me look at me look at me' />

</View>

);

}

}



***Style trong React Native(Đào Duy Phương)***

Tất cả các thành phần cơ bản sẽ sử dụng là một tham số có tên là style. Những tham số style và giá trị của chúng tương tự với cách hoạt động của CSS trên môi trường Web, chỉ có vài khác biệt, ví dụ như backgroundColor thay vì sử dụng background-color trên CSS.

Ví dụ:

import React, { Component } from 'react';

import { StyleSheet, Text, View } from 'react-native';

const styles = StyleSheet.create({

bigBlue: {

color: 'blue',

fontWeight: 'bold',

fontSize: 30,

},

red: {

color: 'red',

},

});

export default class LotsOfStyles extends Component {

render() {

return (

<View>

<Text style={styles.red}>just red</Text>

<Text style={styles.bigBlue}>just bigBlue</Text>

<Text style={[styles.bigBlue, styles.red]}>bigBlue, then red</Text>

<Text style={[styles.red, styles.bigBlue]}>red, then bigBlue</Text>

</View>

);

}

}



***Height & Width(Đào Duy Phương)***

Chiều rộng và dài của các thành phần sẽ xác định kích thước của các thành phần giao diện trên màn hình.

Ví dụ:

class FixedDimensionsBasics extends Component {

render() {

return (

<View>

<View style={{width: 50, height: 50, backgroundColor: 'powderblue'}} />

<View style={{width: 100, height: 100, backgroundColor: 'skyblue'}} />

<View style={{width: 150, height: 150, backgroundColor: 'steelblue'}} />

</View>

);

}

};



***FlexBox(Lê Thanh Liêm)***

Một thành phần có thể chỉ định rõ bố cục cho các view con của nó bằng cách sử dụng thuật toán của flexbox. Flexbox được thiết kế để cung cấp giao diện tương đồng trên những màn hình có kích thước khác nhau.

Ví dụ:

class FlexDirectionBasics extends Component {

render() {

return (

// Try setting `flexDirection` to `column`.

<View style={{flex: 1, flexDirection: 'row'}}>

<View style={{width: 50, height: 50, backgroundColor: 'powderblue'}} />

<View style={{width: 50, height: 50, backgroundColor: 'skyblue'}} />

<View style={{width: 50, height: 50, backgroundColor: 'steelblue'}} />

</View>

);

}

};



1. **Các component cơ bản (Võ Cao Thanh Phong)**
   * Text Component

Một thành phần React để hiển thị văn bản.

Text hỗ trợ lồng, tạo kiểu và xử lý cảm ứng.

*Ví dụ:*

export default class TextInANest extends Component {

constructor(props) {

super(props);

this.state = {

titleText: "Bird's Nest",

bodyText: 'This is not really a bird nest.'

};

}

render() {

return (

<Text style={styles.baseText}>

<Text style={styles.titleText} onPress={this.onPressTitle}>

{this.state.titleText}{'\n'}{'\n'}

</Text>

<Text numberOfLines={5}>

{this.state.bodyText}

</Text>

</Text>

);

}

}

* + TextInput Component

Một thành phần nền tảng để nhập văn bản vào ứng dụng thông qua bàn phím. Đạo cụ cung cấp khả năng cấu hình cho một số tính năng, chẳng hạn như tự động sửa, tự động viết hoa, văn bản giữ chỗ và các loại bàn phím khác nhau, chẳng hạn như bàn phím số.

*Ví dụ:*

export default class PizzaTranslator extends Component {

constructor(props) {

super(props);

this.state = {text: ''};

}

render() {

return (

<View style={{padding: 10}}>

<TextInput

style={{height: 40}}

placeholder="Type here to translate!"

onChangeText={(text) => this.setState({text})}

value={this.state.text}

/>

<Text style={{padding: 10, fontSize: 42}}>

{this.state.text.split(' ').map((word) => word && '🍕').join(' ')}

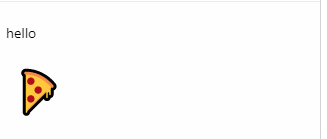
</Text>

</View>

);

}

}



* + Button Component

Một thành phần nút cơ bản sẽ hiển thị độc đáo trên bất kỳ nền tảng nào. Hỗ trợ mức độ tùy biến tối thiểu.

*Ví dụ:*

<View>

<Button

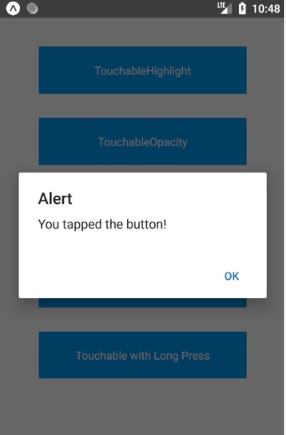
title="Press me"

color="#f194ff"

onPress={() => Alert.alert('Button with adjusted color pressed')}

/>

</View>



* + CheckBox

 Được sử dụng khi một phần tử đại diện cho hộp kiểm có thể được kiểm tra, bỏ chọn hoặc có trạng thái kiểm tra hỗn hợp.

Ví dụ:

import { CheckBox } from 'react-native-elements'

<CheckBox

title='Click Here'

checked={this.state.checked}

/>

<CheckBox

center

title='Click Here'

checked={this.state.checked}

/>

<CheckBox

center

title='Click Here'

checkedIcon='dot-circle-o'

uncheckedIcon='circle-o'

checked={this.state.checked}

/>

<CheckBox

center

title='Click Here to Remove This Item'

iconRight

iconType='material'

checkedIcon='clear'

uncheckedIcon='add'

checkedColor='red'

checked={this.state.checked}

/>

<CheckBox

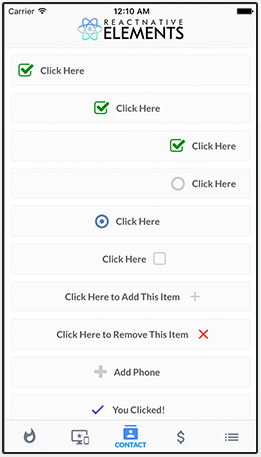
checkedIcon={<Image source={require('../checked.png')} />}

uncheckedIcon={<Image source={require('../unchecked.png')} />}

checked={this.state.checked}

onPress={() => this.setState({checked: !this.state.checked})}

/>



* + Radio Button

Được sử dụng để đại diện cho một nút chọn giá trị của nút đó.

*Ví dụ:*

<Radio onSelect={this.onSelect.bind(this)} defaultSelect={this.state.optionSelected - 1}>

<Option color="gray" selectedColor="#008BEF">

<Item title="First Options" description="This is your First Option"/>

</Option>

<Option color="gray" selectedColor="#008BEF">

<Item title="Second Options" description="This is your Second Option"/>

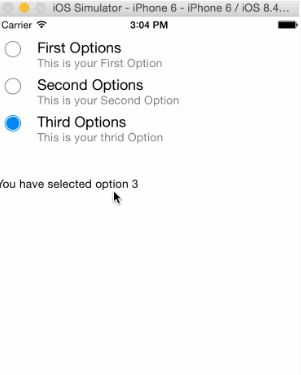
</Option>

<Option color="gray" selectedColor="#008BEF">

<Item title="Third Options" description="This is your thrid Option"/>

</Option>

</Radio>



1. **Các component nâng cao(Đào Duy Phương, Lê Thanh Liêm)**
   * ListView

ListView - Một thành phần cốt lõi được thiết kế để hiển thị hiệu quả các danh sách cuộn theo chiều dọc của dữ liệu thay đổi. API tối thiểu là tạo một [ListView.DataSource](https://facebook.github.io/react-native/docs/listviewdatasource.md), điền vào đó một mảng dữ liệu phẳng và khởi tạo một ListViewthành phần với nguồn dữ liệu đó và một renderRowcuộc gọi lại lấy một đốm từ mảng dữ liệu và trả về một thành phần có thể kết xuất được.

*Ví dụ:*

class MyComponent extends Component {

constructor() {

super();

const ds = new ListView.DataSource({rowHasChanged: (r1, r2) => r1 !== r2});

this.state = {

dataSource: ds.cloneWithRows(['row 1', 'row 2']),

};

}

render() {

return (

<ListView

dataSource={this.state.dataSource}

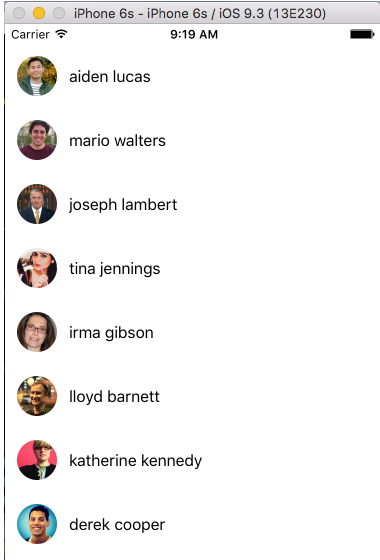
renderRow={(rowData) => <Text>{rowData}</Text>}

/>

);

}

}



* + Picker

Hiển thị thành phần bộ chọn gốc trên iOS và Android

*Ví dụ:*

import React, { Component } from 'react';

import { View, Text, Picker, StyleSheet } from 'react-native'

class PickerExample extends Component {

state = {user: ''}

updateUser = (user) => {

this.setState({ user: user })

}

render() {

return (

<View>

<Picker selectedValue = {this.state.user} onValueChange = {this.updateUser}>

<Picker.Item label = "Steve" value = "steve" />

<Picker.Item label = "Ellen" value = "ellen" />

<Picker.Item label = "Maria" value = "maria" />

</Picker>

<Text style = {styles.text}>{this.state.user}</Text>

</View>

)

}

}

export default PickerExample

const styles = StyleSheet.create({

text: {

fontSize: 30,

alignSelf: 'center',

color: 'red'

}

})

* + Switch Button

Một Switch là một thành phần kiểm soát mà đòi hỏi một callback OnValueChange mà cập nhật value prop để các thành phần để phản ánh các hành động người dùng. Nếu prop giá trị không được cập nhật, các thành phần sẽ tiếp tục làm cho giá trị cung cấp chống đỡ thay vì kết quả mong đợi của bất kỳ hành động sử dụng.

render() {

    return (

      <View style={styles.container}>

        {/\*Text to show the text according to switch condition\*/}

        <Text>{this.state.switchValue?'Switch is ON':'Switch is OFF'}</Text>

        {/\*Switch with value set in constructor\*/}

        {/\*onValueChange will be triggered after switch condition changes\*/}

        <Switch

          style={{marginTop:30}}

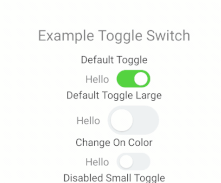
          onValueChange = {this.toggleSwitch}

          value = {this.state.switchValue}/>

      </View>

    );

  }



* + Slider

React-native không dùng slider nữa.

import { Slider } from 'react-native-elements';

<View style={{ flex: 1, alignItems: 'stretch', justifyContent: 'center' }}>

<Slider

value={this.state.value}

onValueChange={value => this.setState({ value })}

/>

<Text>Value: {this.state.value}</Text>

</View>;



* + DrawerLayout

Ngăn kéo (thường được sử dụng để điều hướng) được hiển thị cùng renderNavigationView. Chế độ xem điều hướng ban đầu không hiển thị trên màn hình, nhưng có thể được kéo vào từ bên cạnh cửa sổ được chỉ định bởi drawerPositionprop và chiều rộng của nó có thể được đặt bằng drawerWidthprop.

render: function() {

var navigationView = (

<View style={{flex: 1, backgroundColor: '#fff'}}>

<Text style={{margin: 10, fontSize: 15, textAlign: 'left'}}>I'm in the Drawer!</Text>

</View>

);

return (

<DrawerLayoutAndroid

drawerWidth={300}

drawerPosition={DrawerLayoutAndroid.positions.Left}

renderNavigationView={() => navigationView}>

<View style={{flex: 1, alignItems: 'center'}}>

<Text style={{margin: 10, fontSize: 15, textAlign: 'right'}}>Hello</Text>

<Text style={{margin: 10, fontSize: 15, textAlign: 'right'}}>World!</Text>

</View>

</DrawerLayoutAndroid>

);

},

* + ViewPager

ViewPager được coi là trình quản lý bố cục cho phép người dùng có thể thao tác lật trái và lật phải tương ứng trên màn hình để có thể chuyển qua lại dữ liệu.

render() {

return (

<ViewPagerAndroid

style={styles.viewPager}

initialPage={0}>

<View style={styles.pageStyle} key="1">

<Text>First page</Text>

</View>

<View style={styles.pageStyle} key="2">

<Text>Second page</Text>

</View>

</ViewPagerAndroid>

);

}

* + WebView

Là một**view** giúp cho bạn có thể hiển thị 1 trang web từ URL hoặc là từ 1 trang html tùy chỉnh

import React, {Component} from 'react';

import {WebView} from 'react-native';

class MyWeb extends Component {

render() {

return (

<WebView

source={{uri: 'https://github.com/facebook/react-native'}}

style={{marginTop: 20}}

/>

);

}

}

## *\* Các bài tập component đã tìm hiểu*

* **Võ Cao Thanh Phong**

import React, { Component } from 'react';

import {

StyleSheet,

Text,

View,

TouchableOpacity,

Image,

Alert,

ScrollView,

FlatList,

Button

} from 'react-native';

export default class Posts extends Component {

constructor(props) {

super(props);

this.state = {

data: [

{id:1, title: "ANDROID", time:"1 days ago",image:"https://cungdev.com/wp-content/uploads/2018/02/bg\_android.png"},

{id:2, title: "Beautifull",time:"2 minutes ago",image:"https://lorempixel.com/400/200/nature/5/"} ,

{id:3, title: "Dipiscing elit. Aenean ", time:"3 hour ago",image:"https://lorempixel.com/400/200/nature/4/"},

{id:4, title: "Commodo ligula eget dolor.", time:"4 months ago",image:"https://lorempixel.com/400/200/nature/6/"},

{id:5, title: "Sport",time:"5 weeks a go", image:"https://lorempixel.com/400/200/sports/1/"},

{id:6, title: "Natoque penatibus et magnis", time:"6 year ago",image:"https://lorempixel.com/400/200/nature/8/"},

{id:7, title: "Dis parturient montes, nascetur",time:"7 minutes ago",image:"https://lorempixel.com/400/200/nature/1/"},

{id:8, title: "Ridiculus mus. Donec quam",time:"8 days ago",image:"https://lorempixel.com/400/200/nature/3/"},

{id:9, title: "Felis, ultricies nec, pellentesque",time:"9 minutes ago",image:"https://lorempixel.com/400/200/nature/4/"},

]

};

}

render() {

return (

<View style={styles.container}>

<FlatList style={styles.list}

data={this.state.data}

keyExtractor= {(item) => {

return item.id;

}}

ItemSeparatorComponent={() => {

return (

<View style={styles.separator}/>

)

}}

renderItem={(post) => {

const item = post.item;

return (

<View style={styles.card}>

<View style={styles.cardHeader}>

<View>

<Text style={styles.title}>{item.title}</Text>

<Text style={styles.time}>{item.time}</Text>

</View>

</View>

<Image style={styles.cardImage} source={{uri:item.image}}/>

<View style={styles.cardFooter}>

<View style={styles.socialBarContainer}>

<View style={styles.socialBarSection}>

<TouchableOpacity style={styles.socialBarButton }onPress={()=> alert("like")}>

<Image style={styles.icon} source={{uri: 'https://png.icons8.com/android/75/e74c3c/hearts.png'}}/>

<Text style={styles.socialBarLabel}>999</Text>

</TouchableOpacity>

</View>

<View style={styles.socialBarSection}>

<TouchableOpacity style={styles.socialBarButton}onPress={()=> alert("comment")}>

<Image style={styles.icon} source={{uri: 'https://png.icons8.com/ios-glyphs/75/2ecc71/comments.png'}}/>

<Text style={styles.socialBarLabel}>25</Text>

</TouchableOpacity>

</View>

<View style={styles.socialBarSection}>

<TouchableOpacity style={styles.socialBarButton}>

<Image style={styles.icon} source={{uri: 'https://png.icons8.com/metro/75/3498db/administrator-male.png'}}/>

<Text style={styles.socialBarLabel}>13</Text>

</TouchableOpacity>

</View>

</View>

</View>

</View>

)

}}/>

</View>

);

}

}

const styles = StyleSheet.create({

container:{

flex:1,

marginTop:20,

},

list: {

paddingHorizontal: 17,

backgroundColor:"#E6E6E6",

},

separator: {

marginTop: 10,

},

/\*\*\*\*\*\*\*\* card \*\*\*\*\*\*\*\*\*\*\*\*\*\*/

card:{

shadowColor: '#00000021',

shadowOffset: {

width: 2

},

shadowOpacity: 0.5,

shadowRadius: 4,

marginVertical: 8,

backgroundColor:"white"

},

cardHeader: {

paddingVertical: 17,

paddingHorizontal: 16,

borderTopLeftRadius: 1,

borderTopRightRadius: 1,

flexDirection: 'row',

justifyContent: 'space-between',

},

cardContent: {

paddingVertical: 12.5,

paddingHorizontal: 16,

},

cardFooter:{

flexDirection: 'row',

justifyContent: 'space-between',

paddingTop: 12.5,

paddingBottom: 25,

paddingHorizontal: 16,

borderBottomLeftRadius: 1,

borderBottomRightRadius: 1,

},

cardImage:{

flex: 1,

height: 150,

width: null,

},

/\*\*\*\*\*\*\*\* card components \*\*\*\*\*\*\*\*\*\*\*\*\*\*/

title:{

fontSize:18,

flex:1,

},

time:{

fontSize:13,

color: "#808080",

marginTop: 5

},

icon: {

width:25,

height:25,

},

/\*\*\*\*\*\*\*\* social bar \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

socialBarContainer: {

justifyContent: 'center',

alignItems: 'center',

flexDirection: 'row',

flex: 1

},

socialBarSection: {

justifyContent: 'center',

flexDirection: 'row',

flex: 1,

},

socialBarlabel: {

marginLeft: 8,

alignSelf: 'flex-end',

justifyContent: 'center',

},

socialBarButton:{

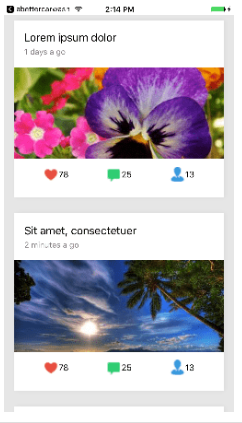
flexDirection: 'row',

justifyContent: 'center',

alignItems: 'center',

}

});



* **Lê Thanh Liêm**

import React, {Component} from 'react';

import {

TouchableOpacity,

StyleSheet,

ScrollView,

View,

Button,

Text,

StatusBar,

} from 'react-native';

export default class App extends Component{

constructor(){

super()

this.state = {

resultText: '',

calculationText: ''

}

this.operations = ['D','+', '-', '\*', '/']

}

calculateResult(){

const text = this.state.resultText

console.log(text, eval(text))

this.setState({

calculationText: eval(text)

})

}

validate(){

const text = this.state.resultText

switch(text.slice(-1)){

case '+':

case '-':

case '\*':

case '/':

return false

}

return true

}

buttonPressed(text){

console.log(text);

if (text == '='){

return this.validate() && this.calculateResult();

}

this.setState({

resultText: this.state.resultText + text

})

}

operate(operation){

switch(operation){

case 'D':

console.log(this.state.result)

let text = this.state.resultText.split('')

text.pop()

this.setState({

resultText: text.join('')

})

break

case '+':

case '-':

case 'x':

case '/':

const lastchar = this.state.resultText.split('').pop()

if(this.operations.indexOf(lastchar) > 0) return

if (this.state.resultText == "" ) return

this.setState({

resultText: this.state.resultText + operation

})

}

}

render() {

let elems = []

let nums = [[1,2,3], [4,5,6], [7,8,9], ['.',0,"="]]

for (let i = 0; i < 4 ; i++){

let row = []

for (let j = 0; j < 3;j++){

row.push(<TouchableOpacity key={nums[i][j]} onPress={() => this.buttonPressed(nums[i][j])} style={styles.btn}>

<Text style = {styles.btnText}>{nums[i][j]}</Text>

</TouchableOpacity>)

}

elems.push(<View key={i} style={styles.row}>{row}</View>)

}

let ops = []

for (let i = 0; i < 5; i++){

ops.push(<TouchableOpacity key={this.operations[i]} style={styles.btn} onPress={() => this.operate(this.operations[i])}>

<Text style={styles.btnText}>{this.operations[i]}</Text>

</TouchableOpacity>)

}

return (

<View style = {styles.container}>

<View style = {styles.result}>

<Text style = {styles.resultText}>{this.state.resultText}</Text>

</View>

<View style = {styles.calculation}>

<Text style = {styles.calculationText}>{this.state.calculationText}</Text>

</View>

<View style = {styles.buttons}>

<View style = {styles.number}>

{elems}

</View>

<View style = {styles.operation}>

{ops}

</View>

</View>

</View>

);

}

}

const styles = StyleSheet.create({

container:{

flex: 1

},

row:{

flexDirection:'row',

flex: 1,

justifyContent: 'space-around',

alignItems:'stretch'

},

btnText:{

fontSize: 25

},

btn:{

flex: 1,

alignItems: 'center',

alignSelf: 'stretch',

justifyContent: 'center'

},

result: {

flex: 2,

backgroundColor: 'black',

justifyContent:"center",

alignItems:"flex-end"

},

resultText:{

fontSize:20,

color:'white'

},

calculation:{

flex: 1,

backgroundColor:'black',

justifyContent:"center",

alignItems:"flex-end"

},

calculationText:{

fontSize: 20,

color: 'white',

},

buttons:{

flex: 7,

flexDirection: 'row'

},

number:{

flex: 3,

backgroundColor:'gray',

color:'white'

},

operation:{

flex: 1,

color:'white',

backgroundColor:'brown',

justifyContent: 'space-around',

alignItems:'stretch'

}

});

* **Đào Duy Phương**

*Gồm 3 file*:

* + *File 1*:

import React, {Component} from 'react';

import { SearchBar } from 'react-native-elements';

import {FlatList,View,Text,TextInput,Image} from 'react-native';

import Icon from 'react-native-vector-icons/MaterialIcons';

import Data from '../search/data/danhsach';

export default class App extends Component {

constructor(prop)

{

super(prop);

this.initData= Data;

this.state={

data: this.initData

};

}

renderItem = ({item}) =>

{

return <View style={{margin :10,flex:1,flexDirection:"row"}}>

<Image style={{width: 50, height: 50}}

source={{uri: item.anh}} />

<View style={{flexDirection:"column",flex:1,margin:1}}>

<Text>{item.name}</Text>

<Text>{item.gmail}</Text>

<Text>{item.sdt}</Text>

</View>

</View>

}

render() {

return (

<View >

<View style={{flexDirection:"row"}}>

<TextInput

placeholder="search"

placeholderTextColor="white"

style={{backgroundColor: '#2f363c',height : 50,fontSize : 16,padding:10,color:'white',

borderBottomWidth:0.5,borderBottomColor:'#7d90a0',width :300,margin : 10}}

/>

<View style={{margin :10,padding:10}}>

<Icon name="add" size={30} color="#900" />

</View>

</View>

<FlatList

data={this.state.data}

keyExtractor={(item)=> item.key.toString()}

renderItem={this.renderItem}/>

</View>

);

}

}

* + *File 2*:

import React, {Component} from 'react';

import { SearchBar } from 'react-native-elements';

import {FlatList,View,Text,TextInput,Image,ActivityIndicator,Button} from 'react-native';

import Icon from 'react-native-vector-icons/MaterialIcons';

import SanPham from './phongban';

export default class App extends React.Component {

constructor(props){

super(props);

this.state ={

myData:[]

}

this.testRef = React.createRef();

this.rInput = React.createRef();

}

componentDidMount(){

return fetch('http://172.16.0.119/QLNV/Select\_SanPham.php')

.then((response) => response.json())

.then((data) => {

this.setState({

myData: data

}).done(); });

const r = this.testRef.current;

}

render() {

if(this.state.isLoading){

return(

<View style={{flex: 1, padding: 20}}>

<ActivityIndicator/>

</View>

)

}

const {navigate} = this.props.navigation;

return (

<View >

<View style={{flexDirection:"row"}}>

<TextInput

placeholder="search"

placeholderTextColor="white"

style={{backgroundColor: '#2f363c',height : 50,fontSize : 16,padding:10,color:'white',

borderBottomWidth:0.5,borderBottomColor:'#7d90a0',width :300,margin : 10}}

/>

<View style={{margin :10,padding:10}}>

<Icon name="add" size={30} color="#900" />

</View>

</View>

<FlatList

data={this.state.myData}

key

renderItem = {({item}) =>

<View>

< SanPham ma ={item.MaPB} tenphong = {item.TenPB}></SanPham>

</View>

}

keyExtractor={(item, index) => index.toString()}

/>

<Button

onPress={() => navigate('nhanvien', {name: 'Jane'})}

title="Chi tiet"

/>

</View>

);

}

}

* + *File 3*:

import {createAppContainer} from 'react-navigation';

import {createStackNavigator} from 'react-navigation-stack';

import Login from './login';

import PB from './App';

import chitiet from './DanhSachNV';

const MainNavigator = createStackNavigator({

Home: {screen: Login},

phongban:{screen: PB},

nhanvien:{screen: chitiet}

});

const App = createAppContainer(MainNavigator);

export default App;

**Chương 4: xây dựng ứng dụng**

**Chương 5: kết luận và kiến nghị**

Tài liệu tham khảo

***\*Cách xử lý mảng trong ES6 (Võ Cao Thanh Phong)***

***\*Cách sử dụng Map trong ES6 (Lê Thanh Liêm)***

***\*Cách sử dụng Set trong ES6 (Đào Duy Phương)***