http://192.168.1.11:8081/

Link học bản 0.59
https://archive.reactnative.dev/docs/getting-started
https://github.com/huongnguyenvan/react-native

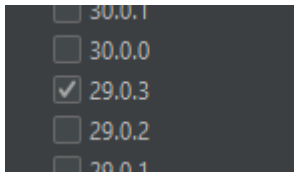
Cài Đặt

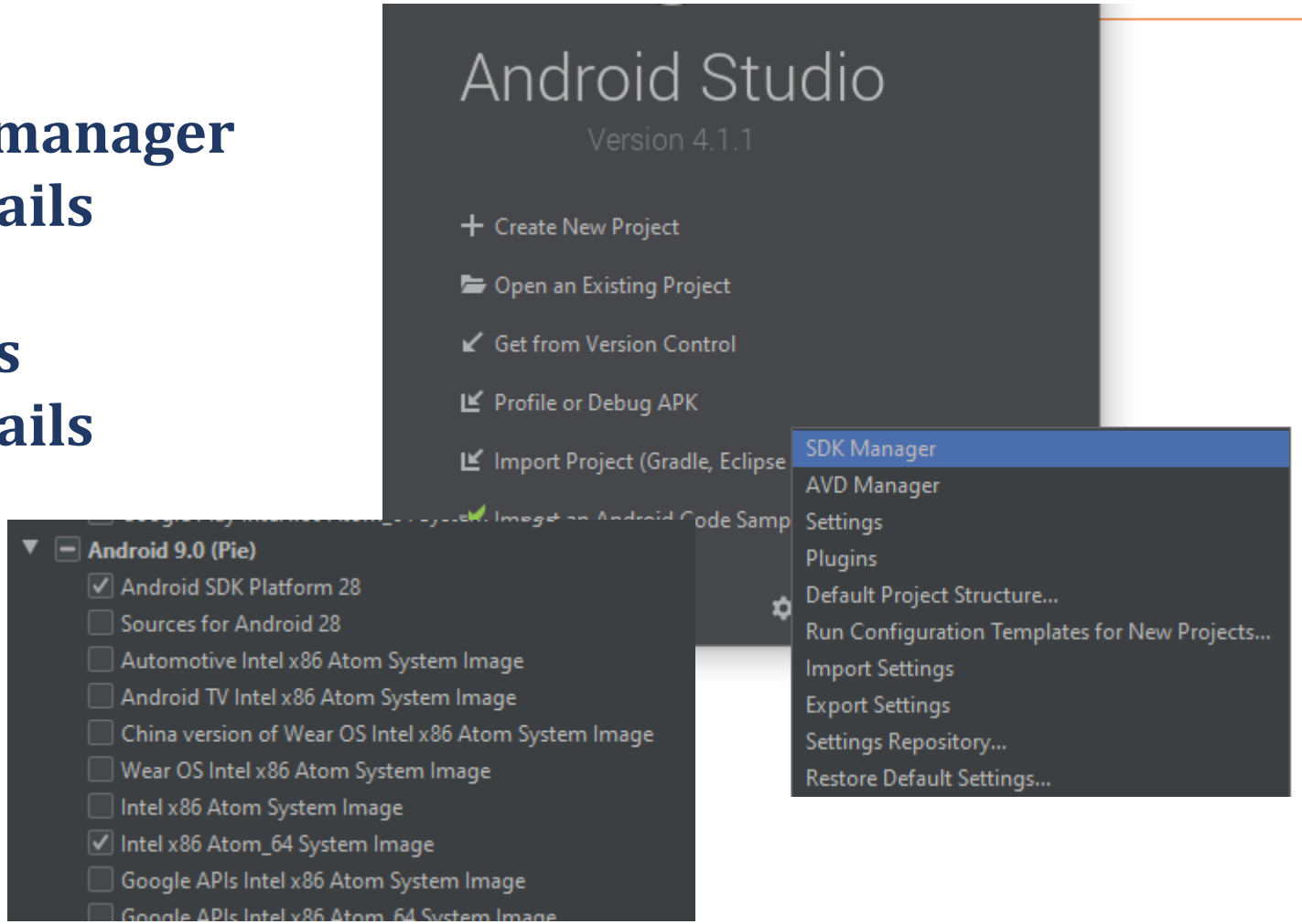
Cài choco trước: vào trang này copy paste vào cmd
+ https://docs.chocolatey.org/en-us/choco/setup#install-with-cmdexe
Hoặc link sau: https://jcutrer.com/windows/install-chocolatey-choco-windows10

Vào Cmd chạy dưới quyền admin:
+ choco install -y nodejs.install python2 jdk8
+ npm install –g react-native-cli

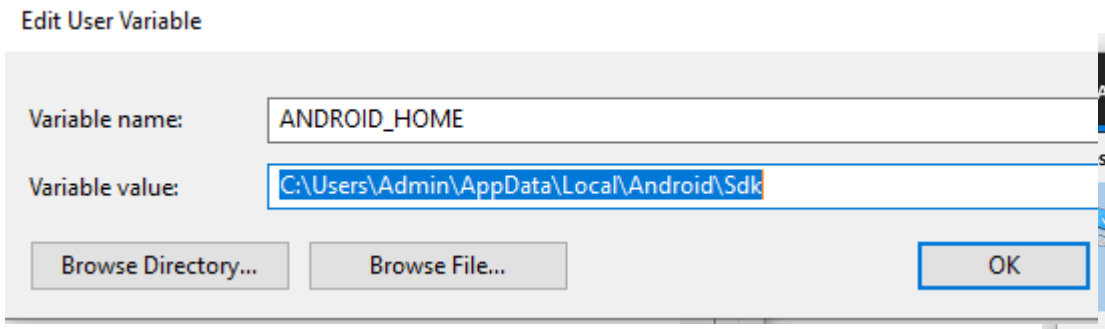
Cài Đặt

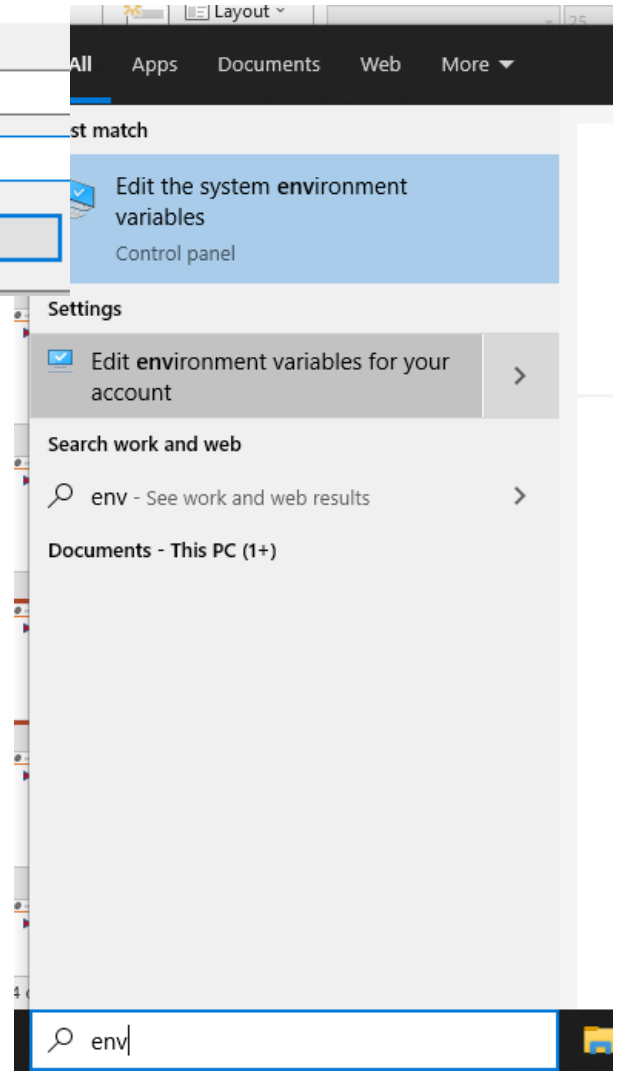
+ vào config -> SDK manager
+ Show Package Details
+ Chọn như hình
+ Sang tab SDK Tools
+ Show Package Details
+ Chọn như hình



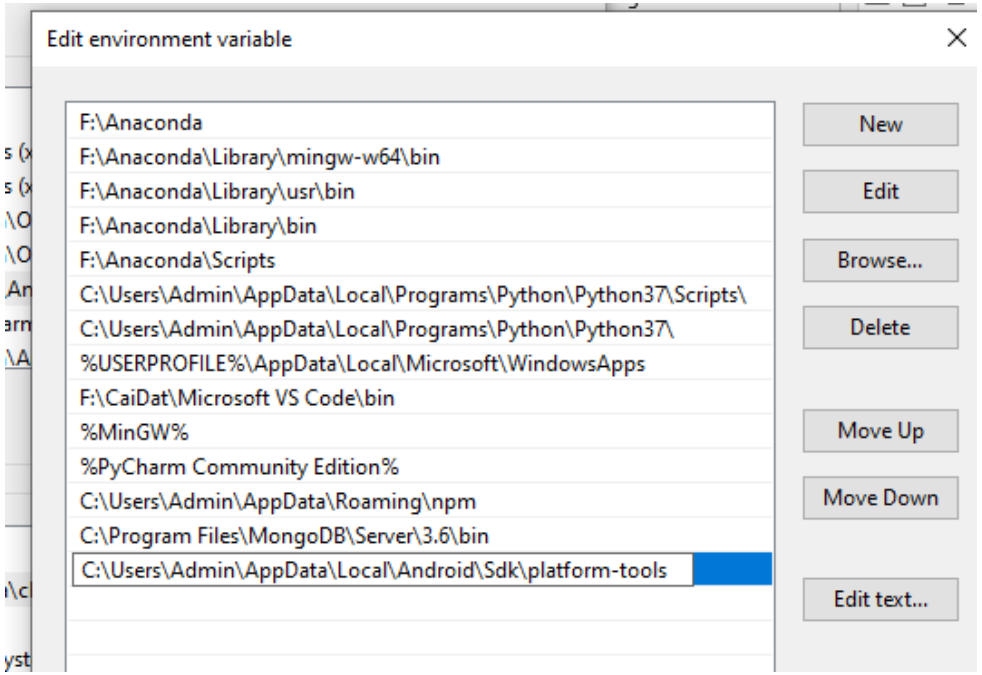


Cài Đặt





Vào EnvirmentVaribale
+ New và them như hình
+ vào Path và them như hình



Cài Đặt

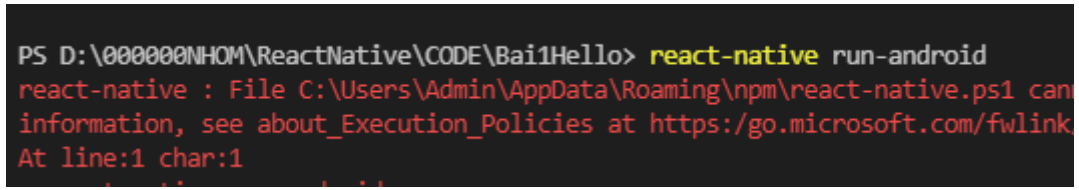
Create Project

+ react-native init NameProject

+ CD vào folder

+ Muốn chạy thì

react-native run-android



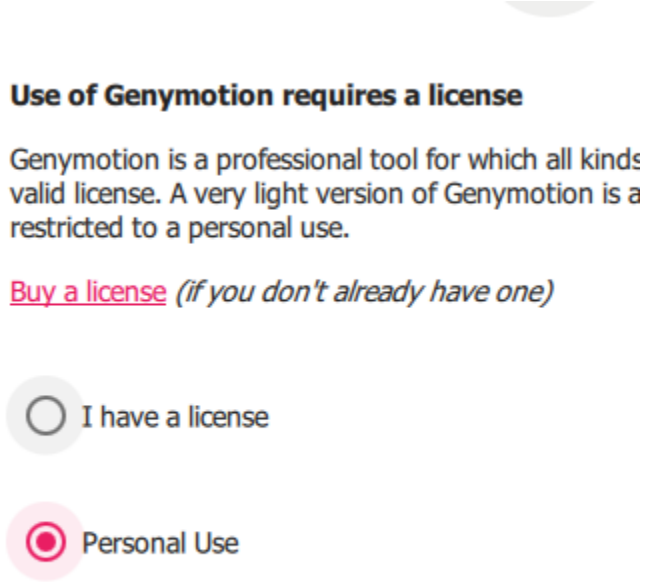
+ Nếu bị lỗi thì bấm (gg lòi mồn mới ra)

Set-ExecutionPolicy Unrestricted -Scope CurrentUser

Cài Đặt

Download máy ảo Gennymotion
+ Vào https://www.genymotion.com/download/
+ nhớ tạo tài khoản

Không thì dung máy thật



Cài Đặt

+ cài các extension
+ https://viblo.asia/p/mot-so-extension-visual-studio-code-huu-ich-cho-react-native-QpmleEkolrd

+ React Native Tools
+ Babel JavaScript
+ Flow Language Support
+ ESLint
+ Prettier — JavaScript formatter

Biến hàm

Biến let chỉ tồn tại trong dấu cặp ngoặc thôi
+ Nên hay dung cho vòng for

```
Let Vantho= () =>{return(abc)}

Let a = [1,2,3,4,5]
a.map((value)=>console.log(value))
```

Phân rã

```
Let nguoi={
Ten:”van tho”,
Tuoi:”22
}

Let {ten:tencuaban,tuoi:tuoicuaban} = nguoi
Console.log(tencuaban+ tuoicuaban)
```

Truyền nhiều tham số

```
Let tinhtong =(a,...nThamSo)
{
        let tong=0;
        nThamSo.map((value)=> tong+=value);
        tong +=a;
        return tong;
}
Console.log(tinhtong(1,2,3,4,5,6));

// cái nthamso nó sẽ là cái mảng
```

Tất cả class đều phải khai báo trong constructor

Hàm filter

Tìm trong mảng tất cả đối tượng thủa mãn yêu cầu

Let data= Mang.filter((value)=>{return value.ten===“Tho”})

Data là 1 mảng thỏa mã tất cả đối tượng có ten=“Tho”

Vòng đời

Vòng đời Component gồm 3 thành phần chính

- **Mounted: Được mở (Mới vào)**
- **Update : có thay đổi(đang ở game thì mở lại app của ta)(vào lại)**
- **UnMounted: Bấm vô home rồi tắt đi (Thoát hẳn đi)**

Mounted

Mounted: các phương thức sẽ chạy theo thứ tự

- **Constructor()**
- **ComponentWillMount()**
- **Render()**
- **componentDidMount(): sau render() và chỉ gọi 1 lần duy nhất**

Update

Update : khi người dung thoát ra và vào lại app của ta (sử dụng 1,3)

1. componentWillReceiveProps(onject nextProps) : Cha của component này sẽ truyền vào một props mới và phương thức này sẽ khởi tạo lại giao diện . Chúng ta có thể cập nhật lại state nội bộ thông qua phương thức this.setState() trước khi phương thức render được gọi.
2. shouldComponentUpdate(object nextProps, object nextState) : Phương thức này có giá trị trả về là kiểu boolean. Một Component có thể được render hoặc không được render, phương thức này nhằm đảm bảo component sẽ được render lại nếu như trả về true. Chúng ta thường dùng phương thức này để kiểm tra props hoặc state có thay đổi hay không nếu như có thay đổi thì chúng ta trả ra true để chạy tiếp render() và trả ra false để ko chạy phương thức render().
3. render() : phương thức này chỉ được gọi lại khi shouldComponentUpdate trả ra true.

Cài Đặt

UnMounting:

- componentWillUnmount() : Phương thức này chỉ xảy ra khi ứng dụng bị đóng hoàn toàn.

Props state

+ Props dung bố truyenf cho con
+ State là cờ , kiểu vậy

Chú ý, luôn có 1 cái view trong return của render của class Component sẽ đc xép xếp theo thứ tự

Props

App.js

```
Bai2Props > App.js > App > render
1   import React, { Component } from 'react';
2   import Ahihi from './Ahihi';
3   import {
4     Text,
5     View
6   } from "react-native"
7
8   class App extends Component {
9     render() {
10      return (
11        <View>
12          <Text > Nguyễn Văn Thọ 15</Text>
13          <Ahihi ten = "Van tho"></Ahihi>
14        </View>
15      );
16    }
17  }
18
19  export default App;
```

Ahihi.js

```
Bai2Props > Ahihi.js > Ahihi > render
1   import React, { Component } from 'react';
2   import {
3       Text,
4       View
5     } from "react-native"
6   class Ahihi extends Component {
7       render() {
8           return (
9           <View>
10            <Text> Tên là  {this.props.ten}</Text>
11          </View>
12          );
13      }
14  }
15  export default Ahihi;
16
```

State và bấm

App.js

Bai3State > App.js > App > ThayDoi

```
1  import React, { Component } from 'react';
2  import{
3    Text,
4    View
5  } from "react-native"
6  class App extends Component {
7    constructor(props) {
8      super(props);
9      this.state={
10       ten:"Nguyễn Văn Thọ",
11       tuoi:22,
12       count:0
13     }
14   }
15   ThayDoi = () =>
16    {
17     this.state.count = !this.state.count;
18     console.log(this.state.count);
19     if(this.state.count)
20     {
21      this.setState({
22        ten:"Nguyễn văn A"}
23      );
24     }
25     else{
26      this.setState({
27        ten:"Nguyễn văn B"}
28      );
29     }
30
31    }
32   render() {
33     return (
34       <View>
35           <Text onPress={()=>this.ThayDoi()}>{this.state.ten}</Text>
36           <Text> tuổi là {this.state.tuoi}</Text>
37       </View>
38     );
39   }
40 }
```

Style

Có tất cả các thuộc tính của CSS
+ Khác nhau: css background-color -> backgroundColor

Có 2 cách
+ Cách 1: dung <Text style={{backgroundColor:while}} >
+ Cách 2: Dùng style ở ngoài và đưa vào component

Style 20

App.js

Bai4Style > App.js > App

```
1   import React, { Component } from 'react';
2   import {
3     Text,
4     View,
5     StyleSheet
6   } from "react-native"
7   class App extends Component {
8     render() {
9       return (
10        <View>
11          <Text style={{backgroundColor:"#2e818a",color:"#e67e22"}}>Nguyễn Văn Thọ</Text>
12          <Text style={css.mauText}>18CDT1</Text>
13        </View>
14      );
15    }
16  }
17  const css = StyleSheet.create({
18    mauText:{
19      backgroundColor:"#d07d1b",
20      fontSize:30,
21      color:"#2ecc71"
22    },
23    mauMoi:{
24      fontSize:40
25    }
26  });
27
28  export default App;
```

Width Hight Flex

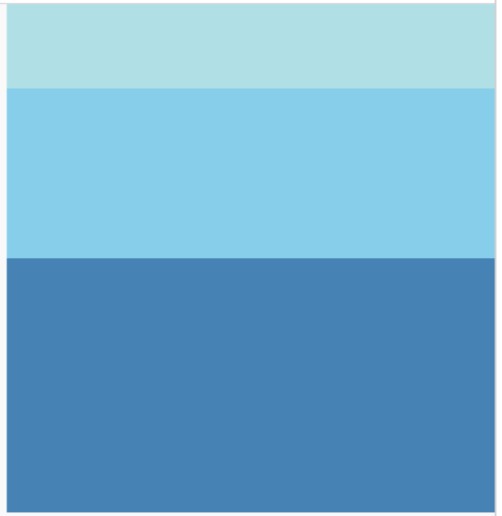
Không cần khai báo kích thước, nó dùng theo điểm ảnh

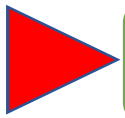
```
import React, { Component } from 'react';
import { View } from 'react-native';

export default 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>
    );
  }
}
```

Flex

Tự động trải đều kích thước còn lại
+ thường khai báo flex:1 để kích thước của nó lấp đầy khaongr trống còn lại
+ 1 component chỉ lấp đầy khoảng trống khi thằng cha nó có kích thước lướn hơn 0
+ đa màn hình
+ flef rồi thì k dung width và height đc, vì width và height ưu tiên hơn
+ view tổng luôn cho flex:1

```
import React, { Component } from 'react';
import { View } from 'react-native';

export default class FlexDimensionsBasics extends Component {
  render() {
    return (
      // Try removing the `flex: 1` on the parent View.
      // The parent will not have dimensions, so the children
can't expand.
      // What if you add `height: 300` instead of `flex: 1`?
      <View style={{flex: 1}}>
        <View style={{flex: 1, backgroundColor: 'powderblue'}} />
        <View style={{flex: 2, backgroundColor: 'skyblue'}} />
        <View style={{flex: 3, backgroundColor: 'steelblue'}} />
      </View>
    );
  }
}
```





FlefBox

Giúp bố trí vị trí component
Có 3 cái chính là :

1. **flexDirection**
2. **justifyContent**
3. **alignItems**

flexDirection

1. Quy định hướng hiển thị của các Component cpn
2. Có 2 thuộc tính và colum và row

Flex Direction

```
import React, { Component } from 'react';
import { View } from 'react-native';

export default 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>
    );
  }
};
```

justifyContent

- **Khai báo sự phân bố của các component con theo trục tọa độ chính**
- **Flex-start, flex-end, flex-center, space-around, space-between**

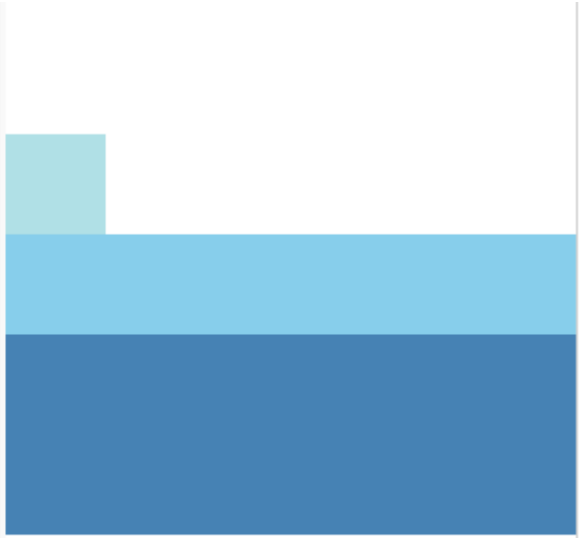
Justify Content

```
import React, { Component } from 'react';
import { View } from 'react-native';

export default class JustifyContentBasics extends Component {
  render() {
    return (
      // Try setting `justifyContent` to `center`.
      // Try setting `flexDirection` to `row`.
      <View style={{
        flex: 1,
        flexDirection: 'column',
        justifyContent: 'space-between',
      }}>
        <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>
    );
  }
};
```

alignItem

Căn theo trục tọa độ phụ
+ nếu khai báo colum rồi thì alignItem sẽ theo trục cow
+ nếu flexDirection khai báo row thì alignItem sẽ theo colum
+ flex-start, flex-end, flex-center

```
export default class AlignItemsBasics extends Component {
  render() {
    return (
      // Try setting `alignItems` to 'flex-start'
      // Try setting `justifyContent` to `flex-end`.
      // Try setting `flexDirection` to `row`.
      <View style={{
        flex: 1,
        flexDirection: 'column',
        justifyContent: 'center',
        alignItems: 'stretch',
      }}>
        <View style={{width: 50, height: 50, backgroundColor:
'powderblue'}} />
        <View style={{height: 50, backgroundColor: 'skyblue'}}
/>
        <View style={{height: 100, backgroundColor:
'steelblue'}} />
      </View>
```



Chia layer

```
class App extends Component {
  render() {
    return (
      <View style={css.bo}>
        <View style={css.con}></View>
      </View>
    );
  }
}
var css = StyleSheet.create({
  bo:{
    flex:1,
    backgroundColor:"red",
    position:'relative'
  },
  con:{
    width:50,
    height:50,
    backgroundColor:"yellow",
    position:'absolute',
    top:50,
    left:100
  }
})
```

- **ellipsizeMode enum (string) : dài quá nên cắt bớt và them bằng dấu (...) , trong đó "string" gồm các giastrij truyền vào là:**

1. **Head: ...abc**
2. **Middle : abc...xyz**
3. **Tail: abc...**
4. **Clip: cắt mất đi luôn, chỉ có trên IOS ☺**
5. **eliipsizeMode chỉ hoạt động khi khai báo cùng với props numOfLine**
6. **numberOfLines number (int): quy định số dòng mà đoạn text hiển thị**
7. **Onlayout function(): đc khởi tạo ở Mount và khi layout thay đổi sẽ phát sinh event là object chưa thông số liên quan tới kích thước của component**

Text

```
<View style={css.container}>
    <Text ellipsizeMode={"tail"} numberOfLines={1} style={css.text}> Go ahead and play around
</View>
```

- numberOfLines number (int) : Quy định số dòng cho phép mà đoạn text được hiển thị.
- onLayout function () : Được khởi tạo ở Mount và khi layout thay đổi. Sẽ phát sinh event là object chứa thông số liên quan tới kích thước của Component. `{nativeEvent: {layout: {x, y, width, height}}}`

- Ví dụ : onLayout((event)=> event.nativeEvent.layout.x) .

- onLongPress function () : Sự kiện nhấn giữ vào component của người dùng.
- onPress function(): Sự kiện nhấn component của người dùng.

•style : sẽ bao gồm các thuộc tính.

Tên thuộc tính	Giá trị	Mô tả
Color	Color (mã màu)	Gán mày cho đoạn text
fontFamily	String	Quy định font chữ
fontSize	Number	Kích thước đoạn text
fontStyle	Enum ('normal','italic')	Kiểu đoạn text
fontWeight	Enum('normal','bold',100->900)	In đậm text
lineHeight	Number	Chiều cao dòng đoạn text
textAlign	Enum('auto','left','center','justify')	Canh lề đoạn text
textDecorationLine	enum('none', 'underline', 'line-through', 'underline line-through')	Bố trí đường gạch ngang cho text

Các thuộc tính props của Text

Tên thuộc tính	Giá trị	Mô tả
textShadowColor	Color (mã màu)	Màu đổ đóng cho text
textShadowOffset	{width: number, height: number}	Kích thước đổ bóng
textShadowRadius	Number	Bo góc cho đổ bóng
textAlignVertical	enum('auto', 'top', 'bottom', 'center')	Canh lề text (android)
fontVariant	[enum('small-caps', 'oldstyle-nums', 'lining-nums', 'tabular-nums', 'proportional-nums')]	Thay đổi font chữ (ios)
letterSpacing	number	Khoảng cách cho text (ios)
textDecorationColor	color	Màu dấu gạch cho text (ios)
textDecorationStyle	enum('solid', 'double', 'dotted', 'dashed')	style đường gạch ngang cho text (ios)

Text

```
<View style={css.container}>
    <Text ellipsizeMode={"tail"} numberOfLines={1} style={css.text}> Go ahead and play around with
    <Text onPress={()=>this._click()} style={css.text1}>Click Here !</Text>
    <Text onLongPress={()=>this._LauLau()} style={css.text1}>Click Lâu</Text>
    <Text selectable={true} style={css.text1}>Go ahead and play around with the Button</Text>
</View>
```

- testID : kiểu String được sử dụng để xác định vị trí kết thúc của đoạn text.
- selectable : kiểu boolean (android). Thường được dùng để kiểm tra người dùng có chọn đoạn text hay không (thường thì làm chức năng copy paste).
- adjustsFontSizeToFit : kiểu boolean (ios). Bằng true sẽ tự động scale kích thước của font bằng với container,
- minimumFontScale : kiểu number (ios). Quy định kích thước nhỏ nhất có thể scale cho font.
- suppressHightlight : kiểu boolean (ios). Nếu bằng true thì sẽ bỏ đi hightlight của text khi người dùng nhấn vào text.

Textinput

- TextInput là một component của React Native cho phép người dùng nhập nội dung văn bản và xử lý các đoạn văn bản đã nhập bởi người dùng.
- TextInput thông thường sẽ đi với 3 sự kiện sau đây : onChangeText, onSubmitEditting và onFocus
- Ví dụ :

```
render() {
  return (
    <TextInput
      style={{height: 40, borderColor: 'gray', borderWidth: 1}}
      onChangeText={(text) => this.setState({text})}
      value={this.state.text}
```

Textinput

•Mặc định TextInput sẽ có đường viền ở dưới và màu của đường viền này được cung cấp bởi hệ thống và nó sẽ không thể bị thay đổi. Chúng ta có thể sử dụng thuộc tính underlineColorAndroid={transparent} để định màu trong suốt cho đường viền đó.

•Để thay đổi hành vi của bàn phím chúng ta có thể sử dụng thuộc tính windowSoftInputMode với tham số là adjustResize

Textinput

•autoCapitalize enum(String) : Tự động viết hoa cho đoạn text. Sẽ nhận vào các tham số sau :
•- characters : tất cả ký tự
•- words : Chữ đầu tiên của mỗi từ
•- sentences : Từ đầu tiên của một câu (mặc định)
•- none : không tự động viết hoa.
•autoCorrect bool : Nếu là true thì sẽ bật chức năng auto-correct (mặc định là true);
•autoFocus bool : nếu là true thì TextInput sẽ được focus khi componentDidMount được gọi (mặc định là false).
•blurOnSubmit bool : Nếu là true thì TextInput sẽ mất focus khi được submit. Mặc định giá trị là true.
•editable bool : Nếu là fale không cho người dùng nhập văn bản.

Textinput

- keyboardType enum (String): Khai báo keyboard sẽ được mở. Bao gồm các tham số : default., email-address, numeric, phone-pad, ascii-capable, numbers-and-punctuation, url, number-pad, name-phone-pad, decimal-pad, twitter, web-search.
- Các tham số làm việc trên đa nền tảng : default, numeric, email-address, phone-pad.
- maxLength number: Giới hạn số ký tự được phép nhập.
- multiline bool : Nếu là true đoạn text có thể nhập nhiều dòng.
- placeholder node : nhập vào chuỗi nhắc nhỡ.
- placeholderTextColor color : Quy định màu text cho placeholder.
- returnKeyType enum : Bao gồm các tham số
- Các tham số làm việc trên đa nền tảng : done, go, next, search, send
- Các tham số làm việc trên android : none, previous.

Textinput

- •secureTextEntry bool : Nếu là true thì các đoạn ký tự sẽ thay thế bằng dấu *.
- •selectTextOnFocus bool : Nếu là true thì đoạn text được gán thuộc tính này sẽ tự động focus.
- •selection {start:number, end:number} : sẽ chọn đoạn văn bản tương ứng với vị trí truyền vào.
- •selectionColor color : Màu highlight của textinput..
- •value string : đoạn text mặc định cho inputtext.
- •inlineImageLeft String : nếu được khai báo thì sẽ render hình ảnh bên trái textinput (Android).
- •inlineImagePadding number : canh padding cho hình ảnh inlineImage (Android).
- •numberOfLines number : số dòng được phép hiển thị (Android)
- •returnKeyLabel string : Trả về key của label (Android).
- •underlineColorAndroid color : màu boder bottom textinput.

Cài Đặt

- clearButtonMode enum : Clear button sẽ xuất hiện bên phải của text input. Bao gồm các tham số : never, while-editing, unless-editing, aways (IOS).
- clearTextOnFocus bool : Nếu là true thì đoạn text sẽ bị xóa khi textinput được focus (IOS).
- dataDetectorTypes enum : Bao gồm các tham số : phoneNumber, link, address, calendarEvent, none, all .(IOS)
- enablesReturnKeyAutomatically bool : Nếu là true bàn phím sẽ disables khi không có text và tự động enables khi có text. (IOS).
- keyboardAppearance enum : Khai báo màu của key board bào gồm các tham số : default, light, dark. (IOS).
- selectionState DocimentSelectionState : đây là một trạng thái đáp ứng nhu cầu lựa chọn thông tin văn bản.

TextInput Event

- onBlur function : Được gọi khi bị mất focus.
- onChange function : Được gọi đoạn text thay đổi.
- onChangeText function : Được gọi khi đoạn text thay được và đoạn text được thay đổi sẽ là tham số của hàm.
- onContentSizeChange function : được gọi khi kích thước của text thay đổi và trả ra object {nativeEvent: { contentSize : {width, height } } }.
- onEndEditing function :được gọi khi kết thúc nhập text.
- onFocus function : Được gọi khi đoạn text được focus.
- onLayout function : Được khởi tạo và layout thay đổi {x, y, width, height}
- onScroll function : Được khởi tạo khi nội dung được scroll sẽ trả ra tham số là một object {nativeEvent: { contentOffset: {x, y } } }.
- onSubmitEditing function : Được gọi khi người dùng nhấn nút submit

TextInput

27

```
return (
  <View style={css.container}>
    <TextInput onChangeText={(event)=>this.TextTenChange(event)}
               placeholderTextColor={"red"}
              placeholder={"Nhập vào họ tên"} ></TextInput>
    <TextInput onChangeText={(event)=>this.TextTuoiChange(event)}
               placeholderTextColor={"red"}
              placeholder={"Nhập Tuổi của bạn"} ></TextInput>
    <Text style={css.text} >{this.state.hienthi}</Text>
    <View style={css.giua}>
      <Text onPress={()=>this.getData()} style={css.button}>Đồng ý</Text>
    </View>
  </View>
);
```

```
TextTenChange  = (event) => {
  this.setState({
    ten:event
  });
}
TextTuoiChange  = (event) => {
  this.setState({
    tuoi:event
  });
```

Image

Ảnh Local

```
<Image source={require('./my-icon.png')} />
```

Ảnh mạng

```
// GOOD
<Image source={{uri: 'https://reactjs.org/logo-og.png'}}
       style={{width: 400, height: 400}} />

// BAD
<Image source={{uri: 'https://reactjs.org/logo-og.png'}} />
```

Image

- **onError** function : Được gọi khi hình ảnh load thất bại.
- **onLayout** function : Được gọi khi ứng dụng được Mount.
- **onLoad** function : Được gọi khi hình ảnh được hiển thị lên mình hình.
- **onLoadEnd** function : Được gọi khi quá trình load hình thành công hoặc thất bại.
- **onLoadStart** function : Được gọi khi hình ảnh vừa bắt đầu load
- **onPartialLoad** function : Được gọi khi một phần của hình ảnh được load hoàn tất. (IOS)
- **onProgress** function : Được gọi trong tiến trình download event sẽ bao gồm tham số : {nativeEvent: {loaded, total } } . (IOS)

Image

• **resizeMode** enum (String): Thay đổi kích thước của hình ảnh bao gồm các tham số :
•- cover : Thay đổi kích thước hình ảnh bằng hoặc lớn hơn kích thước của View chứa nó.
•- contain : Thay đổi kích thước hình ảnh bằng hoặc nhỏ hơn kích thước của view (trừ padding).
•- stretch : Căng điều kích thước của hình ảnh. Sẽ là ảnh hưởng tới. Sẽ làm ảnh hưởng tới tỷ lệ của src.
•- repeat : lập lại hình ảnh.
• **source** imageSourcePropType : Source của hình ảnh sẽ là URL hoặc local file.

Các thuộc tính Style của Image

borderColor color : Màu đường viền.
borderRadius number : Bo góc đường viền
borderTopLeftRadius number : Bo góc viền phía trên bên trái
borderTopRightRadius number :Bo góc viền phía trên bên phải
borderWidth number : Chiều rộng của đường viền
opacity number : Độ trong suốt
overflow enum('visible', 'hidden') : ẩn hoặc hiển thị phần bị tràn ra khỏi view.
resizeMode Object.keys(ImageResizeMode) : thay đổi kích thước
tintColor color : Thay đổi màu của hình ảnh

IMAGE

```
return (
  <View style={css.container}>
      <Image   // ảnh nhỏ hơn thì dùng cover hoặc stretch, ảnh to hơn thì dùng contain
        // source={require('./Image/7.jpg')}
        source={{uri:"https://scontent.fdad3-3.fna.fbcdn.net/v/t1.6435-9/p320x320/227942906_36177508
        resizeMode='cover'
        style={{flex:1,height:null,width:null}} // width và heoght = null mới đc
      />
  </View>
);
```

View

Tổng hợp tham số touch của View

•Các sự kiện liên quan tới hành động của người dùng lên view sẽ trả ra các tham số thông dụng sau :

•- nativeEvent :

- •+changedTouches : là một mảng chứa các sự thay đổi của sự kiện touch.
- •+identifier : ID của touch.
- •+locationX : Trả ra vị trí X nơi người dùng chạm vào.
- •+localtionY : Trả ra vị trí Y nơi người dùng chạm vào.
- •+pageX : Trả ra vị trí X nơi người dùng chạm vào, liên quan tới component cha.
- •+pageY : Trả ra vị trí Y nơi người dùng chạm vào, liên quan tới component cha.
- •+target : id của component đang nhận sự kiện touch
- •+timestamp : thời gian touch, thường được sử dụng để tính toán vận tốc
- •+touches : là một mảng chứa các điểm đã chạm trên màn hình.

View

32

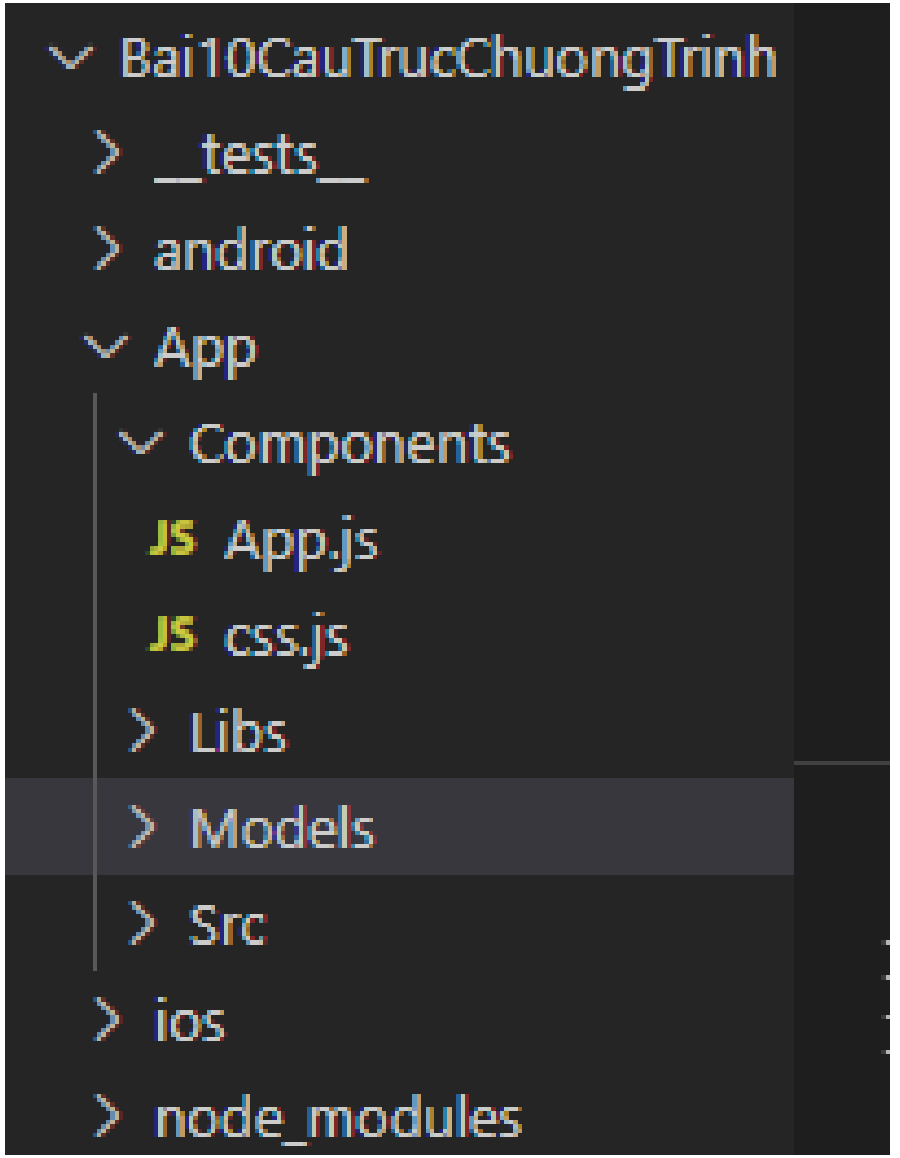
Các thuộc tính props của View

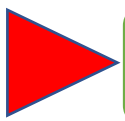
•hitSlop : khai báo thêm phạm vi sẽ nhận sự kiện touch của View. Khuyến cáo nên là 30-40 điểm/mật độ điểm ảnh.
•Ví dụ : Nếu một view có chiều cao là 20 thì sự kiện touch chỉ hoạt động trong khoảng chiều cao đó của View. Nếu như chúng ta muốn mở rộng phạm vi touch vào view lên 40 thì chúng ta sẽ tiến hành khai báo như sau. hitSlop = {{top:10, bottom: 10, left: 0, right: 0 }}.
•Lưu ý : Việc mở rộng phạm vi sẽ bị ảnh hưởng bởi zIndex của View.
•pointerEvents enum (string): Quy định sự kiện touch của View. Bao gồm các tham số :
•- 'auto' : View có thể nhận sự kiện touch.
•- 'none' : View sẽ không nhận sự kiện touch.
•- 'box-one': View sẽ không nhận sự kiện touch nhưng View con thì có thể nhận sự kiện touch.
•Lưu ý : pointerEvent có thể được khai báo trong style

•removeSClippedSubviews bool : Nếu là true thì sẽ cho phép View pháp sinh thanh scroll khi View có nhiều view con vượt ra khỏi màn hình.

Cấu trúc

+ Export thì bỏ vào dấu {}
+ Export default thì import không cần {}
+ Vào index.js sửa lại đường dẫn





Lưu Ý

+ Khai báo mảng ở ngoài Class
+ khi đó bỏ mảng vào render() thì mảng sẽ đc tự duyệt
+ {NameArray}

TouchAble

```
class App extends Component {
    Nut1  = () => {alert("Nút 1");}
    Nut2  = () => {alert("Nút 2");}

    render() {
        return (
            <View style={css.container}>
                <TouchableOpacity
                        onPress={()=>this.Nut1()}
                        style={css.nut}>
                        <Text style={css.text}> Opacity </Text>
                </TouchableOpacity>

                <TouchableHighlight
                        onPress={()=>this.Nut2()}
                        underlayColor={"red"}
                        style={css.nut}>
                        <Text style={css.text}> HighLight </Text>
                </TouchableHighlight>
            </View>
        );
    }
}
```

Navigation

+ Thằng con sẽ mặc định nhận được 2 props là natination và router
+ navigation dung để chuyển màn hình
+ router dung để nhận biến truyền
+ có 3 thuộc tính :
+ navigate : chuyển đến màm hình
+ push : Tạo ra 1 màn hình nữa
+ replate : Thay thế màn hình

Navigation

Router

```
import React, { Component } from 'react';
import { NavigationContainer } from '@react-navigation/native';
import { createNativeStackNavigator } from '@react-navigation/native-stack';
import App from "./App"
import Home from "./Home"
const Stack = createNativeStackNavigator();

class Router extends Component {
    render() {
        return (
            <NavigationContainer>
                <Stack.Navigator initialRouteName="App">
                    <Stack.Screen name="App" component={App} options={{ title: 'Welcome App' }}/>
                    <Stack.Screen name="Home" component={Home} options={{ title: 'Welcome Home' }}/>
                </Stack.Navigator>
            </NavigationContainer>
        );
    }
}

export default Router;
```

Navigation

App

```
class App extends Component {
    constructor(props) {
        super(props);
    }

    GoToHome  = () =>
    {
        console.log("GoToHome");
        this.props.navigation.navigate('Home', { name: 'Thọ' })
    }

    render() {
        return (
            <View style={{flex:1, justifyContent:"center",alignItems:"center"}}>
                <Text onPress={()=>this.GoToHome()} style={{width:100, height:70, fontSize:17}} >GoToHome</Text>
            </View>
        );
    }
}

export default App;
```

Cài Đặt

Home

```
class Home extends Component {
    GoToApp  = () =>
    {
        console.log("GoToApp");
        this.props.navigation.navigate('App')
    }

    render() {
        return (
            <View style={{flex:1, justifyContent:"center",alignItems:"center"}}>
                <Text onPress={()=>this.GoToApp() } style={{width:100, height:70, fontSize:17}}>GoToApp</Text>
                <Text style={{width:100, height:70, fontSize:17}}>Nhận đc là : {this.props.route.params.name}</Text>
            </View>
        );
    }
}

export default Home;
```

Phải sử dụng thư viện ngoài
+ link : https://www.npmjs.com/package/react-native-progress
+ npm install react-native-progress –save

ProgressBar

```
import React, { Component } from 'react';
import * as Progress from 'react-native-progress';
import {
  View
} from "react-native"
class App extends Component {
  render() {
    return (
      <View style={{flex:1, justifyContent:"center",alignItems:'center'}}>
          <Progress.Bar progress={0.3} width={200} />
      </View>
    );
  }
}

export default App;
```

ActivityIndecation

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

class App extends Component {
  render() {
    return (
      <View style={[styles.container, styles.horizontal]}>
        <ActivityIndicator />
        <ActivityIndicator size="large" />
        <ActivityIndicator size="small" color="#0000ff" />
        <ActivityIndicator size="large" color="#00ff00" />
      </View>
    );
  }
}
```

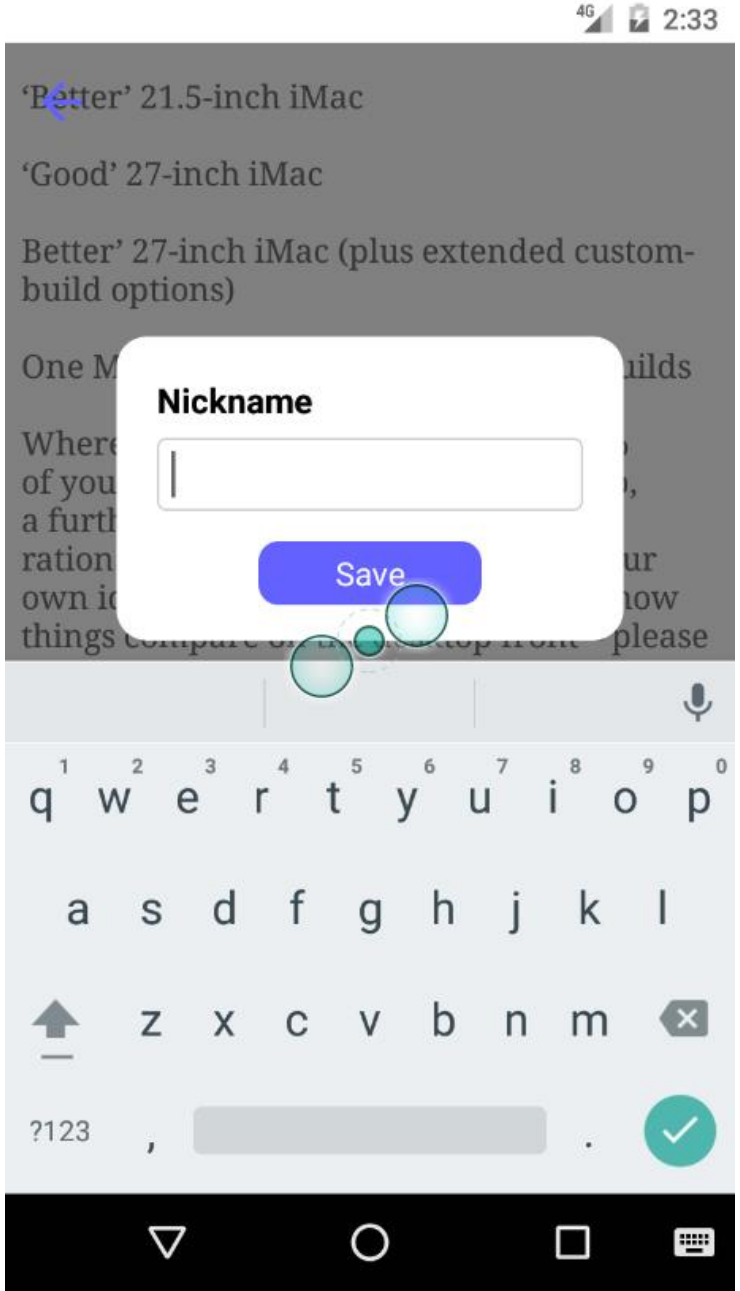
FlatList

```
import { FlatList, StyleSheet, Text, View } from 'react-native';
import React, { Component } from 'react';
class App extends Component {
  click  = (item) =>
  {
    alert(item.key);
  }
  render() {
    return (
      <View style={styles.container}>
        <FlatList
          data={[
            {key: 'Thọ '},
            {key: 'Thanh '},
            {key: 'Tài'},
            {key: 'Bình'},
            {key: 'Đức'},
            {key: 'Dũng '},
            {key: 'Huỳnh'},
          ]}
          renderItem={({item}) => <Text onPress={()=>this.click(item)} style={styles.item}> {item.key}</Text>}
        />
    </View>
```

ScrollView

```
import React, { Component } from 'react';
import {ScrollView,StyleSheet,View} from 'react-native';
class App extends Component {
  render() {
    return (
      <ScrollView style={css.hihi}>
        <View style={css.item}></View>
        <View style={css.item}></View>
        <View style={css.item}></View>
        <View style={css.item}></View>
        <View style={css.item}></View>
        <View style={css.item}></View>
        <View style={css.item}></View>
        <View style={css.item}></View>
        <View style={css.item}></View>
        <View style={css.item}></View>
        <View style={css.item}></View>

      </ScrollView>
    );
  }
}
```

Model


2:33
'Better' 21.5-inch iMac
'Good' 27-inch iMac
Better' 27-inch iMac (plus extended custom-build options)
Nickname
Save

Model

Quên thì lên web học lại ☺

```
    super(props);
    this.state={
      open:true
    }
  }
  NutBam  = () => {this.setState({open:!this.state.open});}
  render() {
    return (
        <View style={css.container}>
            <Modal animationType="slide" transparent={true} visible={this.state.open}>
                <View style={css.model}>
                  <View style={css.modelView}>
                    <Text style={css.text}>Đóng model chứ nhờ !</Text>
                    <TouchableOpacity style={css.giua} onPress={()=>this.NutBam()}>
                        <Text style={css.btn1}> Close</Text>
                    </TouchableOpacity>
                  </View>
                </View>
            </Modal>
            <TouchableOpacity onPress={()=>this.NutBam()}>
                <Text style={css.btn}> Show Model</Text>
            </TouchableOpacity>
        </View>
    );
  }
}
```

Pressable

+ Bắt sự kiện bấm vào hoặc thả ra
+ onPressIn
+ onPressOut
+ onPress

```
class App extends Component {
  bamvao  = () => {alert("Bấm Vào")}
  Thara  = () => {alert("Thả Ra")}

  render() {
    return (
      <View style={css.container}>
          <Pressable onPressIn={()=>this.bamvao()}>
            <Text style={css.text}>Falling</Text>
          </Pressable>
          <Pressable style={css.cach} onPressOut={()=>this.Thara()}>
            <Text style={css.text}>Rising</Text>
          </Pressable>
      </View>
    );
  }
}
```

Picker

Link : https://github.com/react-native-picker/picker

+ npm install @react-native-picker/picker –save

+ Kích vào sổ xuống rồi chọn

```
class App extends Component {
  constructor(props) {
    super(props);
    this.state={
      value:"java"
    }
  }

  render() {
    return (
      <View style={css.container}>
        <View style={css.giua}>
          <Picker
              selectedValue={this.state.value}
              onValueChange={(itemValue, itemIndex) => this.setState({value:itemValue})}>
              <Picker.Item label="Java" value="java" />
              <Picker.Item label="JavaScript" value="js" />
              <Picker.Item label="Python" value="python" />
            </Picker>
        </View>
      </View>

    );
  }
}
```

Slider

Link : https://www.npmjs.com/package/react-native-sliders + npm i --save react-native-sliders

Thanh kéo

```
constructor(props) {
  super(props);
  this.state={
    giatri:0
  }
}

render() {
  return (
    <View style={styles.container}>
      <Slider
        value={this.state.value}
        onValueChange={(value) => this.setState({ giatri:value })}
        minimumValue={0}
        maximumValue={100}
        step={1}
        thumbTintColor={"#e84118"}
        minimumTrackTintColor={"#44bd32"}
      />
      <Text>
        Value: {this.state.giatri}
      </Text>
    </View>
  );
}
}
```

KeyBoar

+ Bình thường keyBoar che đi textInput của ta nên dung cái ni

```
import React, { Component } from 'react';
import { KeyboardAvoidingView, TextInput, StyleSheet } from 'react-native';
class App extends Component {
  render() {
    return (
      <KeyboardAvoidingView behavior={'position'} keyboardVerticalOffset={100}>
            <TextInput placeholder="Username" style={styles.textInput} />
            <TextInput placeholder="PassWord" style={styles.textInput1} />
      </KeyboardAvoidingView>
    );
  }
}
const styles = StyleSheet.create({
  textInput: {
    height: 40,
    borderColor: "#000000",
    borderBottomWidth: 1,
    marginBottom: 10,
    marginTop:400
  },
  textInput1: {
    height: 40,
    borderColor: "#000000",
    borderBottomWidth: 1,
    marginBottom: 10,
  }
});
export default App;
```

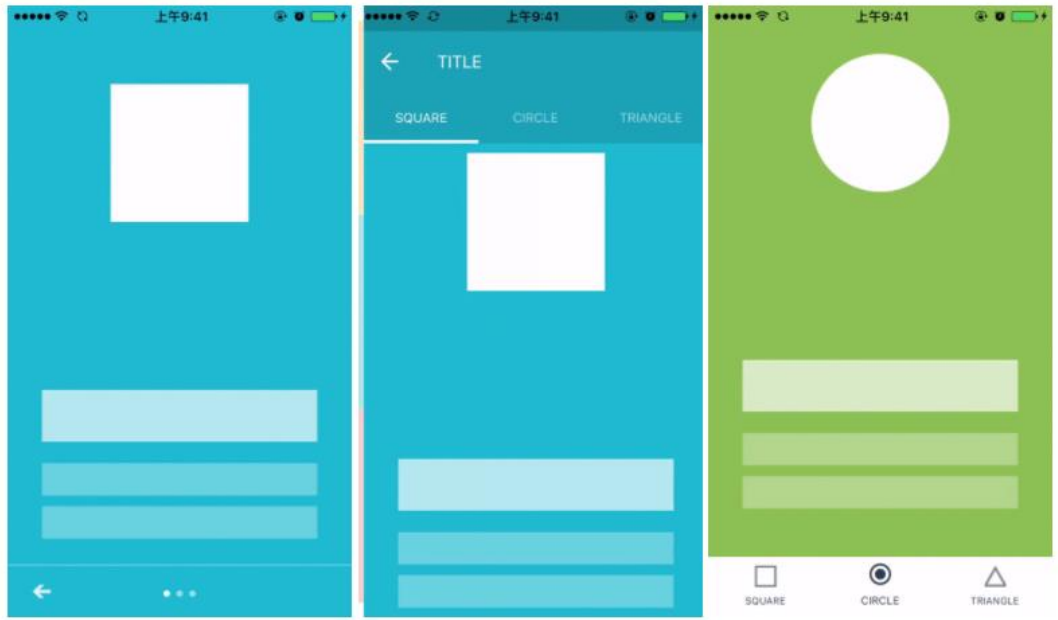
Drawer

```
open  = () => {this.drawer.openDrawer()}
close = () => {this.drawer.closeDrawer()}
render() {
  return (
    <View style={css.container}>
        <DrawerLayoutAndroid  style={css.giua}
        ref={drawer=>this.drawer= drawer}
        renderNavigationView={()=>this.RenderNavigate()}
        drawerWidth={300}
        drawerBackgroundColor={"white"}>
          <View style={css.giua}>
            <TouchableOpacity onPress={()=>this.open()}>
              <Text style={css.btn}>Open</Text>
            </TouchableOpacity>
          </View>
        </DrawerLayoutAndroid>
    </View>
  );
}
}
```

```
class App extends Component {
  RenderNavigate  = () =>
  {
    return (
      <View style={css.container}>
          <View style={css.chia3}>
              <ImageBackground style={css.img}
              resizeMode={'stretch'} source={require("./images/backgroundmeterial.jpg")} >
                  <Text style={css.text}>Nguyễn Văn Thọ</Text>
              </ImageBackground>
          </View>
          <View style={css.chia7}>
            <View style={css.giua}>
              <TouchableOpacity onPress={()=>this.close()}>
                <Text style={css.btn1}>Close</Text>
              </TouchableOpacity>
            </View>
          </View>
      </View>
    );
  }
```

ViewPager

Link: https://github.com/callstack/react-native-pager-view
+ npm install react-native-pager-view



ViewPager

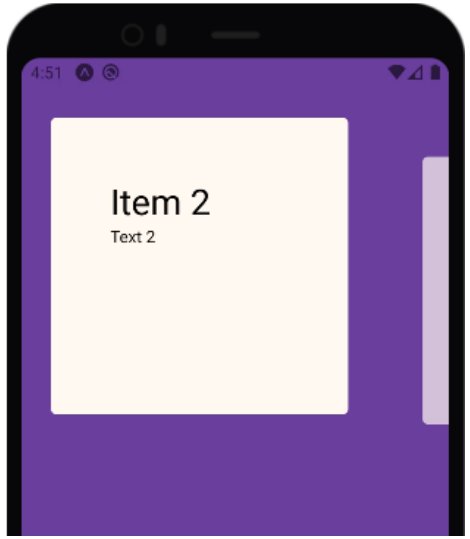
Bai24ViewPagerAndTab > App.js > css

```
1   import React, { Component } from 'react';
2   import { StyleSheet, View, Text } from 'react-native';
3   import PagerView from 'react-native-pager-view';
4   class App extends Component {
5     render() {
6       return (
7         <PagerView style={css.pagerView} initialPage={0}>
8         <View key="0">
9           <Text>Trang 1</Text>
10        </View>
11        <View key="1">
12          <Text>Trang 2</Text>
13        </View>
14        <View key="2">
15          <Text>Trang 3</Text>
16        </View>
17      </PagerView>
18      );
19    }
20  }
21  const css = StyleSheet.create({
22    pagerView: {
23      flex: 1,
24    },
25  });
26  export default App;
```

carousel

Link : https://github.com/meliorence/react-native-snap-carousel?fbclid=IwAR24-RdtQQesoLOa_OB5ONl2sOEW7EHM2y-4Dq0PwRnGYMazp2J-Ur4i9Jc

+ npm install –save react-native-snap-carousel



WebView

Link : https://www.npmjs.com/package/react-native-webview

+ npm i react-native-webview

App.js

Bai26WebView > App.js > default

```
1  import React, { Component } from 'react';
2  import { WebView } from 'react-native-webview';
3  class App extends Component {
4    render() {
5      return (
6        <WebView source={{ uri: 'https://zingmp3.vn/' }} />
7      );
8    }
9  }
10
11 export default App;
```

Alert

App.js

Bai27Alert > App.js > App > showAlert

```
1  import { View, StyleSheet, Button, Alert } from "react-native";
2  import React, { Component } from 'react';
3  class App extends Component {
4   showAlert = () =>
5    Alert.alert(
6      "Alert Title",
7      "My Alert Msg",
8      [
9        {text: "Đồng Ý",onPress: () => Alert.alert("Cancel Pressed"),style: "cancel"},
10       {text: "Hủy",onPress: () => Alert.alert("Cancel Pressed"),style: "cancel"},
11     ]
12   );
13   render() {
14     return (
15       <View style={styles.container}>
16         <Button title="Show alert" onPress={()=>this.showAlert()} />
17       </View>
18     );
19   }
20 }
21
22 const styles = StyleSheet.create({
23   container: {
24     flex: 1,
25     justifyContent: "center",
26     alignItems: "center"
27   }
28 });
29 export default App;
```

Animated

Bai28Animated > App.js > App

```
1   import React, { Component } from 'react';
2   import {
3     View,
4     Text,
5     StyleSheet,
6     Animated
7   } from "react-native"
8   class App extends Component {
9     constructor(props) {
10      super(props);
11      this.state={
12        dichuyen: new Animated.Value(0)
13      }
14    }
15    click  = () =>
16    {
17      Animated.timing(
18        this.state.dichuyen,
19        {
20          toValue:100,
21          duration:1000,
22          useNativeDriver: true
23        }
24      ).start();
25    }
26    render() {
27      return (
28        <View style={css.container}>
29          <Text style={css.btn} onPress={()=>this.click()}>Van Tho</Text>
30          <Animated.View  style={{marginLeft:this.state.dichuyen}}>
31          <Text style={css.btnDiChuyen} >Bị di chuyển</Text>
32          </Animated.View>
33        </View>
34      );
35    }
36  }
```

Check-Box

Link : https://www.npmjs.com/package/react-native-check-box

+ npm i react-native-check-box --save

Check-Box

Bai31CheckBox > App.js > App > Click

```
1   import React, { Component } from 'react';
2   import CheckBox from 'react-native-check-box'
3   import {
4     View,
5     Text
6   } from "react-native"
7   class App extends Component {
8     constructor(props) {
9       super(props);
10      this.state={
11        isCheck:false
12      }
13    }
14    Click  = () =>
15    {
16      this.setState({isCheck:! this.state.isCheck});
17    }
18
19    render() {
20      return (
21        <View style={{flex:1, justifyContent:"center", marginLeft:150}}>
22              <CheckBox
23                onClick={()=>this.Click()}
24                isChecked={this.state.isCheck}
25                rightText={"Chọn"}
26                rightTextStyle={{color:"#ffa502", fontSize:17}}
27                checkedCheckBoxColor={"blue"}
28                uncheckedCheckBoxColor={"green"}
29                // checkedImage={<Image source={require('../../page/my/img/ic_check_box.png
30                // unCheckedImage={<Image source={require('../../page/my/img/ic_check_box_o
31              ></CheckBox>
32        </View>
33      );
34    }
35  }
36
37  export default App;
```

RadioButton

Link : https://www.npmjs.com/package/react-native-simple-radio-button

```
+ npm i react-native-simple-radio-button --save
```

+ Cái return phía trước sẽ là biến truyền vào sau ".then"

Fetch

App.js

Bai32Fetch > App.js > App > getName

```
import React, { Component } from 'react';
import {
  View,
  Text,
  TouchableOpacity,
  StyleSheet
} from "react-native"
class App extends Component {
  constructor(props) {
    super(props);
    this.state={
      name:"Tho",
      i:0,
    }
  }
  getName  = () =>
  {
    this.setState({i:this.state.i+1});
    if(this.state.i === 4) this.setState({i:0});

    myRequest= new Request('https://reactnative.dev/movies.json');
    fetch(myRequest)
    .then((respone)=>respone.json())
    .then((responeJson)=>{
      return responeJson.movies;
    })
    .then((mangMovie)=>{
        this.setState({
          name:mangMovie[this.state.i].title
        })
    }).done();
  }
  render() {
    return (
      <View style={css.container}>
            <Text style={css.ten}>{this.state.name}</Text>
            <TouchableOpacity style={css.btn} onPress={()=>this.getName()}>
              <Text style={css.text}>Get Name movie</Text>
            </TouchableOpacity>
      </View>
```

Async Storage

Link : https://react-native-async-storage.github.io/async-storage/docs/install
+ npm install @react-native-async-storage/async-storage
+ import AsyncStorage from '@react-native-async-storage/async-storage';

Lưu data

Storing string value

```
const storeData = async (value) => {
  try {
    await AsyncStorage.setItem('@storage_Key', value)
  } catch (e) {
    // saving error
  }
}
```

Storing object value

```
const storeData = async (value) => {
  try {
    const jsonValue = JSON.stringify(value)
    await AsyncStorage.setItem('@storage_Key', jsonValue)
  } catch (e) {
    // saving error
  }
}
```

Đọc data

Reading string value

```
const getData = async () => {
  try {
    const value = await AsyncStorage.getItem('@storage_Key')
    if(value !== null) {
      // value previously stored
    }
  } catch(e) {
    // error reading value
  }
}
```

Reading object value

```
const getData = async () => {
  try {
    const jsonValue = await AsyncStorage.getItem('@storage_Key')
    return jsonValue != null ? JSON.parse(jsonValue) : null;
  } catch(e) {
    // error reading value
  }
}
```

Xóa data

removeItem

Removes item for a `key`, invokes (optional) callback once completed.

Signature:

```
static removeItem(key: string, [callback]: ?(error: ?Error) => void): Promise
```

Returns:

`Promise` object.

Example:

```
removeValue = async () => {
  try {
    await AsyncStorage.removeItem('@MyApp_key')
  } catch(e) {
    // remove error
  }

  console.log('Done.')
}
```

Cài Đặt

```
saveData  = async  () =>
{
 try {
   await AsyncStorage.setItem('ten', this.state.ten);
   await AsyncStorage.setItem('tuoi', this.state.tuoi);
   alert("Write Data Success");
 } catch (e) {
   // saving error
 }
}
ReadData  = async  () =>
{
 try {
   const value1 = await AsyncStorage.getItem('ten')
   const value2 = await AsyncStorage.getItem('tuoi')
   if((value1 !== null)  && (value2 !== null)) {
     this.setState({
       layten:value1,
       laytuoi:value2
     });
   }
 } catch(e) {
   // error reading value
 }
}
```

NetInfo

Kiểm tra kết nối Interret hoặc 3G hay Blutooth hay có kết nối không
+ link : https://www.npmjs.com/package/@react-native-community/netinfo
+ npm install --save @react-native-community/netinfo

NetInfo

App.js

Bai34NetInfo > App.js > App > render

```
1   import React, { Component } from 'react';
2   import NetInfo from "@react-native-community/netinfo";
3   import {
4     View,
5     Text,
6     TouchableOpacity,
7     StyleSheet
8   } from "react-native"
9   class App extends Component {
10    constructor(props) {
11      super(props);
12      this.state={
13        ketnoi:"Chưa kết nối",
14        mang:""
15      }
16    }
17    subscribe = NetInfo.addEventListener(data => {
18      if(data.isConnected === true) var x= "Đã kết nối";
19      else x= "Chưa kết nối";
20      this.setState({
21        ketnoi: x,
22        mang: data.type
23      });
24    });
25
26    render() {
27      return (
28        <View style={{flex:1,justifyContent:'center',alignItems:'center'}}>
29            <View style={{marginTop:50, width:200}}>
30              <Text style={css.text}>Mang: {this.state.mang}  </Text>
31              <Text style={css.text}>Trạng thái : {this.state.ketnoi}</Text>
32            </View>
33        </View>
34      );
35    }
36  }
```

Thay đổi Props

+ Tạo 1 view tên myText trong constructor
+ Tạo 1 Obj tên là myStyle, trong đó muốn thay đổi props nào thì điền vào đó
+ Khai báo ID bằng ref và gán view là myText cho cái view khai báo ID đó
+ Sau đó chỉ việc gán cái myStyle cho cái myText là ok
+ Trong cái myStyle gán Props nào vào cũng được(style cũng chỉ là 1 props)

Thay đổi Props

+ Sẽ ghi đè color từ màu cam sang màu vàng của Text

```
class App extends Component {
  constructor(props) {
    super(props);
    myText=null;
    myStyle={
      style:{
        color:'yellow'
      }
    }
  }
  thayDoi  = () =>
  {
      myText.setNativeProps(myStyle);
  }
  render() {
    return (
      <View style={css.container}>
        <Text
            ref={(text)=>myText=text}
            style={css.text}>Nguyễn Văn Thọ</Text>
        <TouchableOpacity onPress={()=>this.thayDoi()}>
          <Text style={css.btn}>Thay đổi CSS</Text>
        </TouchableOpacity>
      </View>
    );
  }
}
var css= StyleSheet.create({
  container:{
    flex:1,
    justifyContent:'center',
    alignItems:'center'
  },
  btn:{
    padding:15,
    backgroundColor:"aqua",
    borderRadius:10,
    color:"orange",
```

Props nào cũng đc

+ Không đc bỏ vào cái view đầu tiên
+ Lấy khoảng cách caaph nhật vào top left là nó di chuyển đc, và top left đó dung state để cập nhật thì hình sẽ di chuyển luôn

PanResponder

```
class App extends Component {
  constructor(props) {
    super(props);
    _HinhTronStyle={}
    this.state={
      pan:new Animated.ValueXY()
    }
  }
  UNSAFE_componentWillMount () {
    _panResponder = PanResponder.create({
      onStartShouldSetPanResponder:(event,gestureState)=>true,  // đăng kí hàm bấm vào
      onMoveShouldSetPanResponder:(event,gestureState)=>true,  // đăng kí hàm di chuyển, luôn có 2 hàm trên
      onPanResponderMove:(event,gestureState)=>{              // thực thi hàm di chuyển
        _HinhTronStyle.style.left = _preLeft + gestureState.dx; // cập nhật giá trị hiện tại đang di chuyển
        _HinhTronStyle.style.top = _preTop + gestureState.dy;   // vì cập nhật state nên cập nhật đến
        this._upDateStyle();                                    // đâu thì hình di chuyển đến đó
      },
      onPanResponderRelease:(event,gestureState)=>{         // thực thi hàm nâng tay lên
        _preLeft += gestureState.dx;                       // cập nhật lại giá trị tọa độ ban đầu
        _preTop += gestureState.dy;
      }
    })
  }
  componentDidMount() {
    this._upDateStyle();
  }

  _upDateStyle(){  // cập nhật lại state
    myView && myView.setNativeProps(_HinhTronStyle)   // kiểm tra myview có null hay không
  }
  render() {
    return (
      <View style={css.container}>
          <View
            ref={(view)=>myView=view}
            style={css.tron}
            {..._panResponder.panHandlers}
          >
          </View>
      </View>
    );
  }
}
var css= StyleSheet.create({
  container:{
    flex:1
  },
  tron:{
    width:80,
    height:80,
    borderRadius:40,
    backgroundColor:"green"
  }
```

DatePicker

+ link: https://github.com/xgfe/react-native-datepicker
+ npm install react-native-datepicker --save

DatePicker

App.js

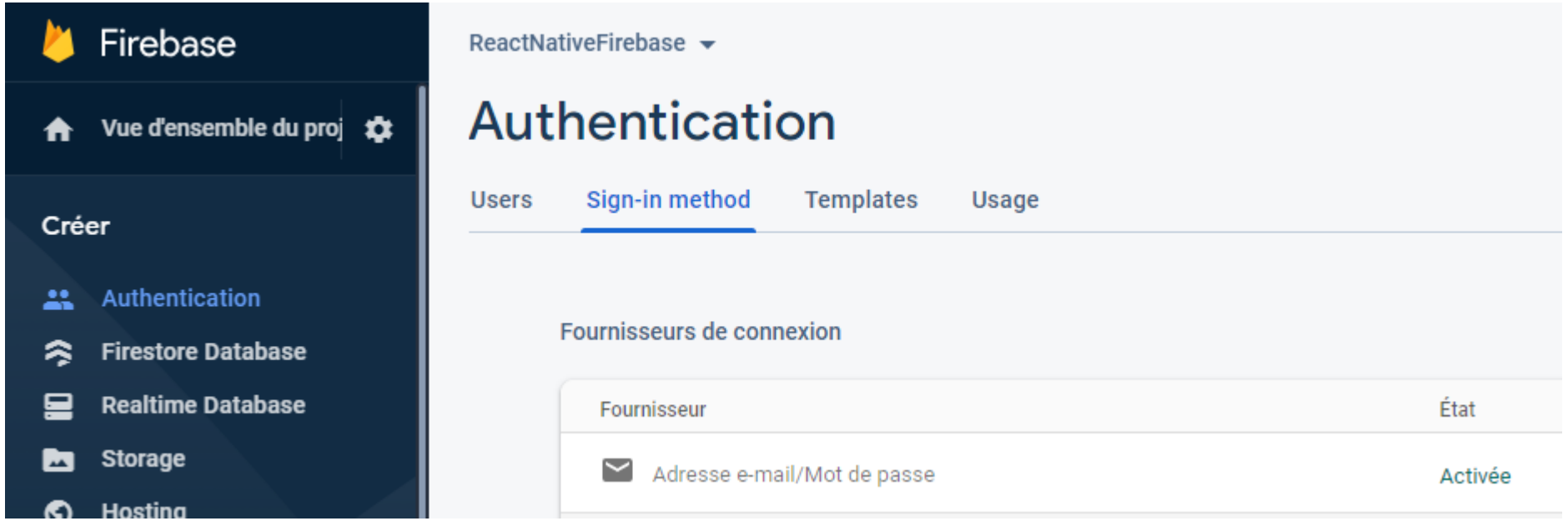
Bai38DatePicker > JS App.js > ...

```
1   import React, { Component } from 'react'
2   import DatePicker from 'react-native-datepicker'
3   export default class App extends Component {
4     constructor(props){
5       super(props)
6       this.state = {date:"2016-05-15"}
7     }
8     render(){
9       return (
10        <DatePicker
11          style={{width: 200}}
12          date={this.state.date}
13          mode="date"
14          placeholder="select date"
15          format="YYYY-MM-DD"
16          minDate="2016-05-01"
17          maxDate="2016-06-01"
18          confirmBtnText="Confirm"
19          cancelBtnText="Cancel"
20          customStyles={{
21            dateIcon: {
22              position: 'absolute',
23              left: 0,
24              top: 4,
25              marginLeft: 0
26            },
27            dateInput: {
28              marginLeft: 36
29            }
30          }}
31          onDateChange={(date) => {this.setState({date: date})}}
32        />
```

FileBase Auth

+ link : https://firebase.google.com/docs/auth/web/password-auth

+ Avtive gmail trong firebase Authentication



FileBase Auth

+ Chọn web

Firebase Auth

+ sẽ ra đc cái này

Utiliser npm Utiliser une balise <script>

Si vous utilisez déjà npm et un bundler de module comme webpack ou Rollup, vous pouvez exécuter la commande suivante pour installer la dernière version du SDK :

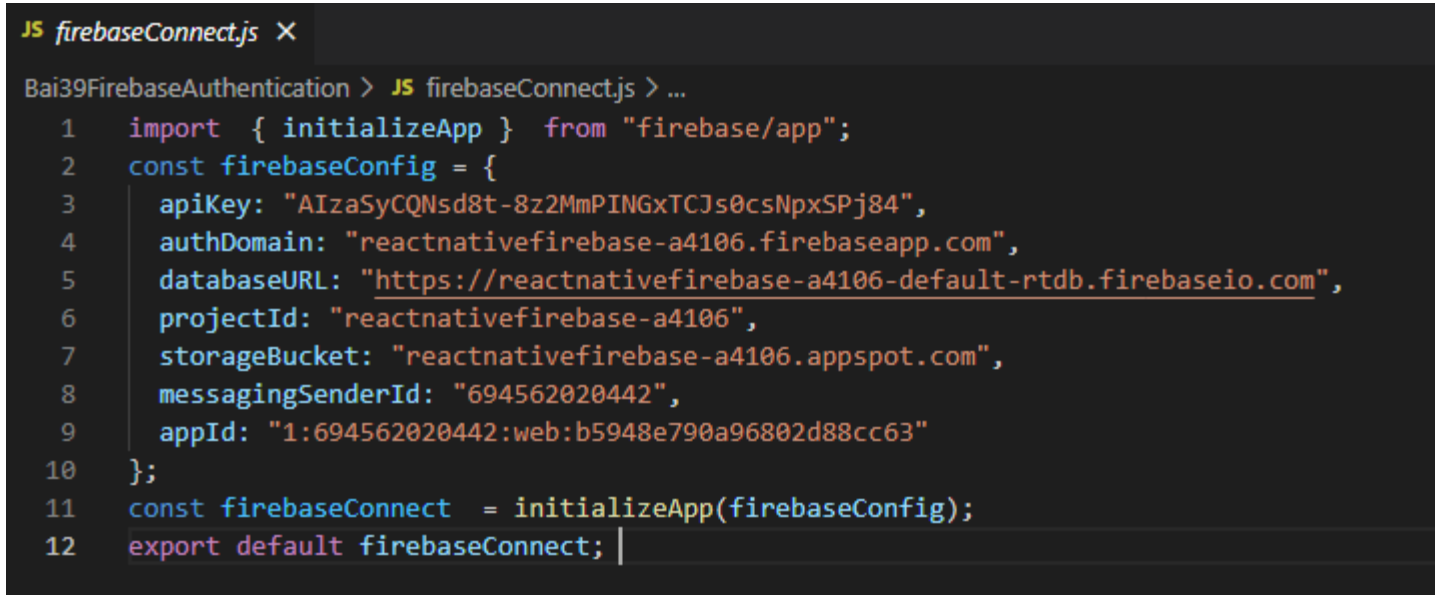
```
$ npm install firebase
```

Ensuite, initialisez Firebase et commencez à exploiter les SDK pour les produits que vous souhaitez utiliser.

```
// Import the functions you need from the SDKs you need
import { initializeApp } from "firebase/app";
// TODO: Add SDKs for Firebase products that you want to use
// https://firebase.google.com/docs/web/setup#available-libraries

// Your web app's Firebase configuration
const firebaseConfig = {
  apiKey: "AIzaSyCQNsd8t-8z2MmPINGxTCJs0csNpxSPj84",
  authDomain: "reactnativefirebase-a4106.firebaseapp.com",
  projectId: "reactnativefirebase-a4106",
  storageBucket: "reactnativefirebase-a4106.appspot.com",
  messagingSenderId: "694562020442",
  appId: "1:694562020442:web:74166628691d0a1488cc63"
};

// Initialize Firebase
const app = initializeApp(firebaseConfig);
```

Firebase Auth

+ npm install firebase
+ tạo 1 file firebaseConnect.js



Firebase Auth

ai39FirebaseAuthentication > JS App.js > ...

```
1  import React, { Component } from 'react';
2  import { getAuth, createUserWithEmailAndPassword } from "firebase/auth";
3  import { signInWithEmailAndPassword } from "firebase/auth";
4  import { signOut } from "firebase/auth";
5  import firebaseConnect from './firebaseConnect'
6  import {
```

```
DangKy  =  () =>
{
    const auth = getAuth();
    createUserWithEmailAndPassword(auth, this.state.user, this.state.pass)
    .then((userCredential) => {
      const user = userCredential.user;
      alert("Đăng Ký Thành Công ");
    })
    .catch((error) => {
      alert("Đăng Ký Thất Bại");
    });
}
```

```
DangNhap  =  () =>
{
 const auth = getAuth();
 signInWithEmailAndPassword(auth, this.state.user, this.state.pass)
    .then((userCredential) => {
      const user = userCredential.user;
      alert("Đăng Nhập Thành Công ");
    })
    .catch((error) => {
      alert("Đăng Nhập Thất Bại");
    });
}
```

```
DangXuat  =  () =>
{
 const auth = getAuth();
 signOut(auth).then(() => {
   alert("Đăng Xuất Thành Công ");
 }).catch((error) => {
   alert("Đăng Xuất Thất Bại");
 });
}
```

Firebase Storage

+ Link : https://firebase.google.com/docs/storage/web/start
+ Thư viện chọn ảnh: https://github.com/baronha/react-native-multiple-image-picker?fbclid=IwAR3Ah8hSumPSDlL4q6KwMgEYvTRWvNaF4PU2RZBttbHFKTC0IyD6todKYvo

RealTime DB

+ Link : https://firebase.google.com/docs/database/web/read-and-write
+ npm install firebase --save

Firebase DB

```
import React, { Component } from 'react';
import firebaseConnect from './firebaseConnect'
import { getDatabase, ref, set , push, onValue, remove} from "firebase/database";
```

```
Them  =  () =>
 {
  // const db = getDatabase();
  // set(ref(db, 'MyDB/' + "ID2"), {
  //   Ten: "Nguyễn Văn B",
  //   Tuoi: 99,
  // })
  // .then(() => {
  //   alert("Thêm thành công");
  // })
  // .catch((error) => {
  //   alert("Thêm thất bại");
  // });
  const db = getDatabase();
  push(ref(db, 'MyDB/'), {
    Ten: "Nguyễn Văn C",
    Tuoi: 99,
  })
  .then(() => {
    alert("Thêm thành công");
  })
  .catch((error) => {
    alert("Thêm thất bại");
  });
 }
```

```
Xem  =  () =>
{
 const db = getDatabase();
 const starCountRef = ref(db, 'MyDB/');
 onValue(starCountRef, (snapshot) => {
   var arrayData=[];
   snapshot.forEach(element=>{
     const key = element.key;
     const Ten = element.val().Ten;
     const Tuoi = element.val().Tuoi;
     arrayData.push({
       key:key,
       Ten:Ten,
       Tuoi:Tuoi
   })
   })
   console.log(arrayData);
 });
}
```

```
Xoa  =  () =>
{
 const db = getDatabase();
 remove(ref(db, 'MyDB/'+ "ID2"))
 .then(() => {
   alert("Xoa thành công");
 })
 .catch((error) => {
   alert("Xoa thất bại");
 });
}
```


A

+