## DAY 1

## Function Component và Class Component trong React Native

Trong React Native, có hai loại thành phần chính để xây dựng giao diện người dùng: Function Component và Class Component. Chúng ta sẽ điểm qua từng loại và so sánh chúng để bạn có cái nhìn rõ ràng về cách chọn loại nào phù hợp với dự án của mình.

****Function Component****

Function Component là một cách đơn giản và dễ đọc để tạo thành phần trong React Native. Chúng ta sử dụng hàm JavaScript thông thường để định nghĩa Function Component:

****

**Ưu điểm của Function Component bao gồm:**

1. Dễ đọc và dễ hiểu: Cú pháp gọn gàng hơn so với Class Component
2. Tích hợp hooks: React cung cấp hooks như `useState()`, `useEffect()`,... cho Function Component, giúp quản lý trạng thái và hiệu suất một cách dễ dàng.
3. Hiệu năng tốt hơn: Thường nhẹ hơn so với Class Component vì không có nhiều phức tạp trong quá trình rendering.

**Class Component**

Class Component là một phần cũ của React Native và vẫn được sử dụng trong nhiều dự án hiện tại. Đây là cách tạo Class Component:

****

****Ưu điểm của Class Component bao gồm:****

1. Cơ hội sử dụng các tính năng cũ: Nếu bạn đã làm việc với React trước khi hooks được giới thiệu, Class Component cung cấp một cách để sử dụng các tính năng cũ như `componentDidMount`, `componentDidUpdate`,...
2. Kế thừa và mở rộng: Class Component cho phép bạn kế thừa và mở rộng từ các lớp khác, giúp quản lý mã nguồn lớn hơn.
3. Làm việc với redux và navigation: Nhiều thư viện phổ biến như Redux và React Navigation sử dụng Class Component cho tích hợp dễ dàng**.**

**So Sánh và Lựa Chọn**

**Khi so sánh Function Component và Class Component trong React Native, nên xem xét các yếu tố sau:**

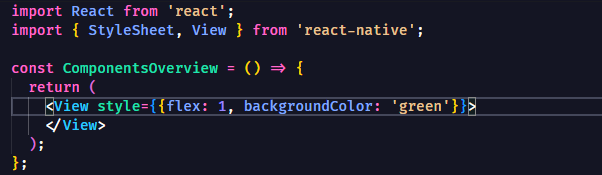
* Độ phức tạp của dự án: Dự án đơn giản thường sử dụng Function Component. Đối với các dự án lớn và phức tạp hơn, Class Component có thể hữu ích.
* Hiệu năng: Function Component thường có hiệu năng tốt hơn vì không có overhead từ các phương thức mà Class Component có. Nếu hiệu năng là ưu tiên, Function Component là lựa chọn tốt.
* Thư viện và tích hợp: Nếu bạn sử dụng các thư viện hoặc tích hợp có sẵn với Class Component hoặc bạn cảm thấy thoải mái với cách Class Component làm việc, hãy sử dụng chúng.
* Quản lý trạng thái: Function Component thích hợp cho việc quản lý trạng thái với hooks. Class Component sử dụng `setState` và các phương thức thời gian thực thi.

Trong nhiều trường hợp, Function Component được ưa chuộng hơn vì sự đơn giản và hiệu suất tốt. Tuy nhiên, quyết định cuối cùng phụ thuộc vào yêu cầu cụ thể của dự án và sự thoải mái của bạn trong việc sử dụng cả hai loại.

## Các Component Cơ Bản

****1. View****

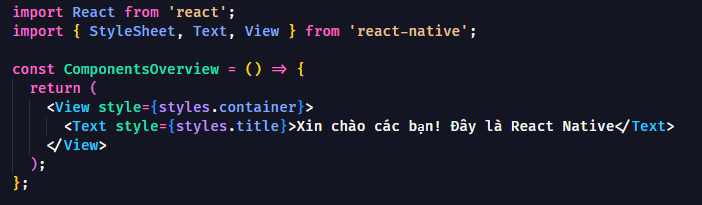
****`View` l****à một container cho phép bạn nhóm các thành phần khác lại với nhau và quản lý vị trí, kích thước, và các thuộc tính khác của chúng, tương tự với thẻ <div> trong HTML. Đây là một ví dụ cơ bản:

********

****Hãy viết code lại như hình ở trên.****

****2. Text****

****`Text`**** là thành phần dùng để hiển thị văn bản, tương tự như thẻ <p> hoặc <span> trong HTML. Bạn có thể sử dụng Text để hiển thị các đoạn văn bản, tiêu đề, nút chọn và nhiều nội dung khác.

****

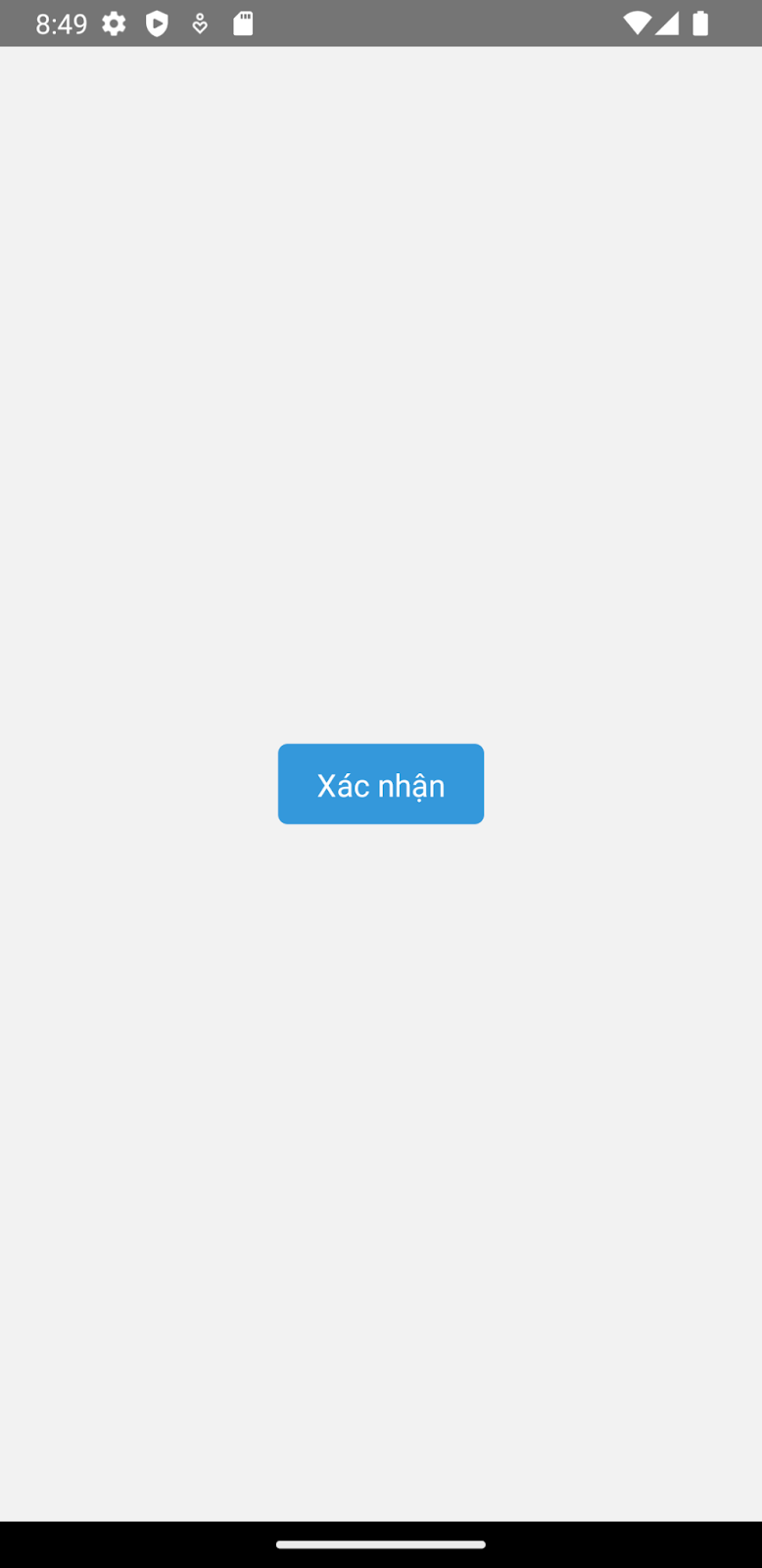
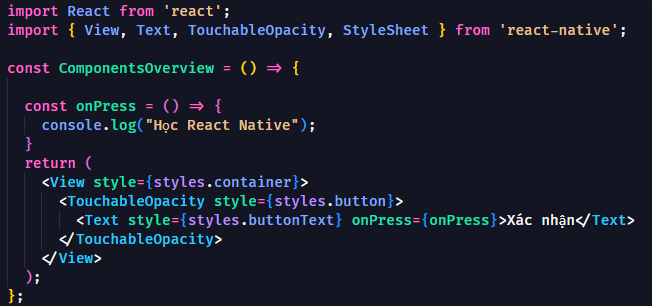
**Hãy viết code lại như hình ở trên.**

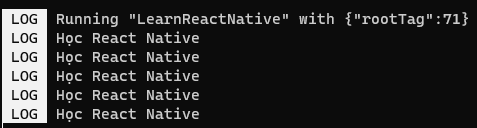
 3. TouchableOpacity

****`TouchableOpacity`****là một thành phần chứa nội dung có thể nhấn được, tương tự thẻ <button> trong HTML. Khi người dùng nhấn vào nó, nó sẽ thay đổi trạng thái và thực hiện các hành động tương ứng, các thuộc tính chính trong TouchableOpacity như:

‘onPress’: sự kiện khi nhấn vào nút nhấn (tương tự có onPressIn, onPressOut,...)

‘style’: style css cho nút nhấn





****Bài tập 2: Tạo một màn hình với thành phần TouchableOpacity với sự kiện log ra màn hình tên, lớp, mssv****

4. Image

****`Image`**** là thành phần cho phép bạn hiển thị hình ảnh trong ứng dụng của bạn, tương tự với thẻ <img> trong HTML, bao gồm các thành phần chính như:

****‘source’:****là đường dẫn tới hình ảnh cần hiển thị

****‘style’:****style css cho hình ảnh





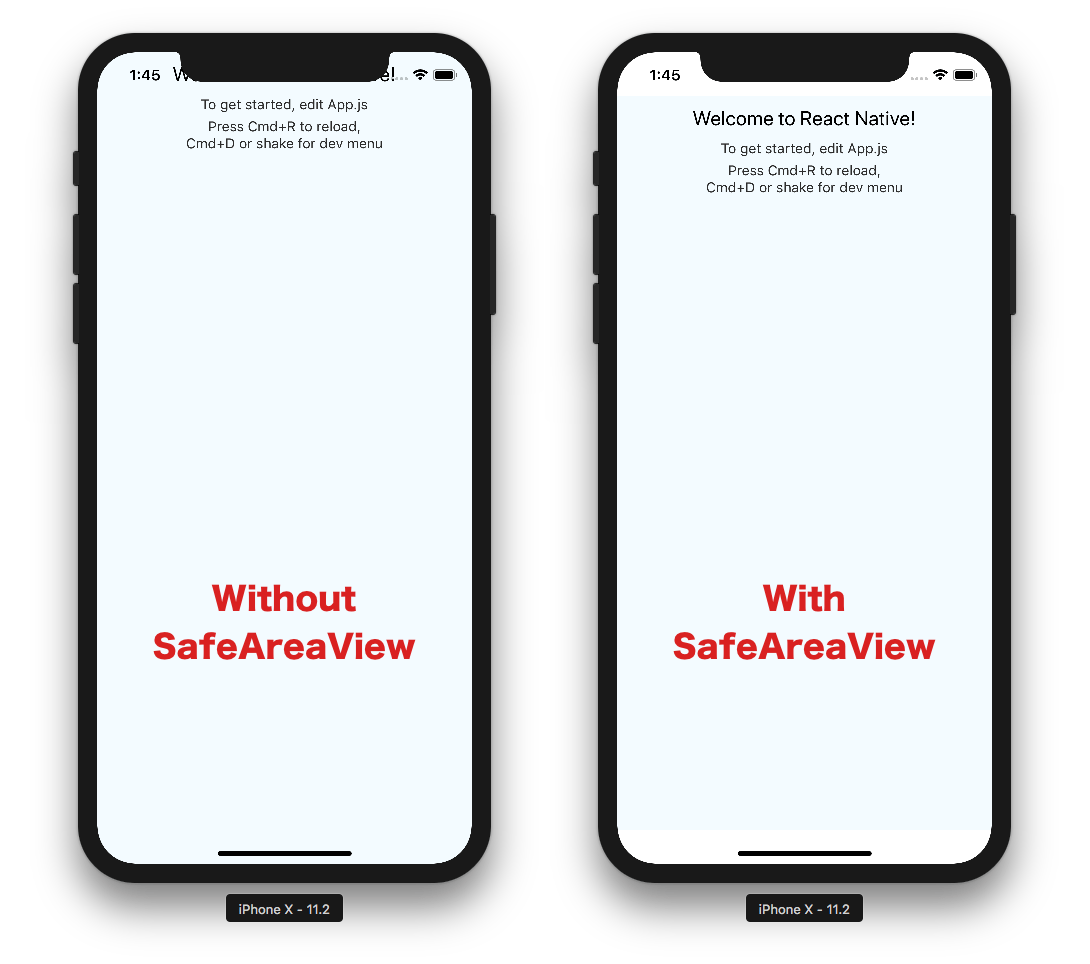
****Bài tập 3: Tạo một màn hình với các component đã học ở trên hiển thị hình ảnh bất kì ở giữa màn hình.****

## Các Component Nâng Cao

****1. SafeAreaView****

****`SafeAreaView`****giúp đảm bảo rằng nội dung của bạn không bị che khuất bởi thanh tiêu đề hoặc thanh dưới cùng của thiết bị. Thường được sử dụng trong các thiết bị có thanh tiêu đề chắn màn hình như iphone và một số thiết bị khác

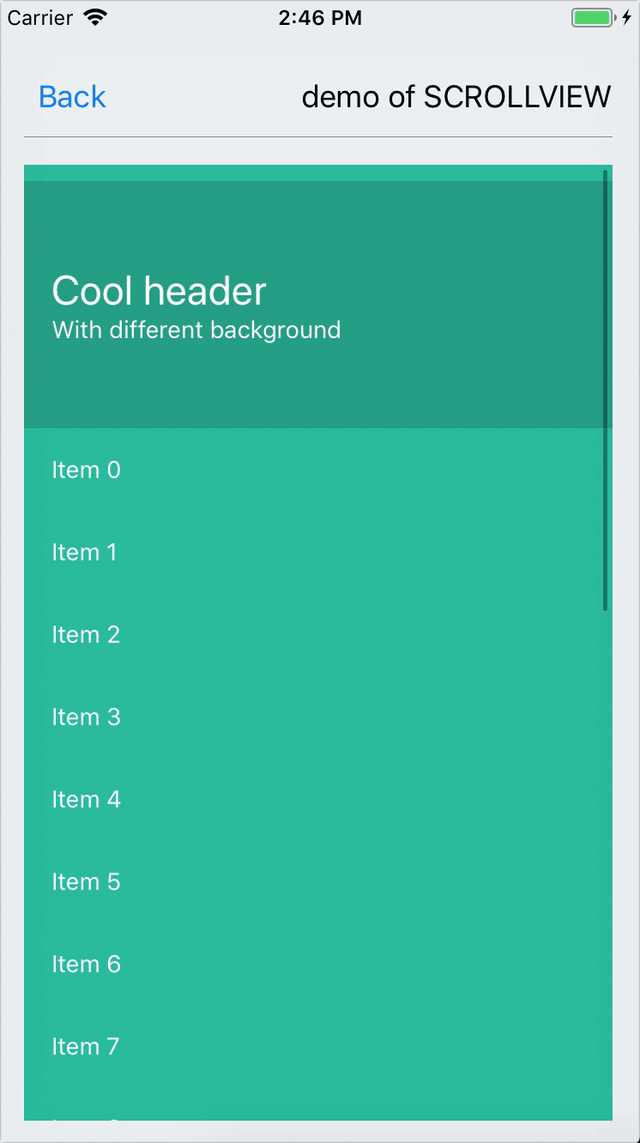




****2. List Views và ScrollView****

React Native cung cấp****`FlatList` và `ScrollView`****cho việc hiển thị danh sách và nội dung cuộn. Có thể tìm hiểu thêm tại****:****[FlatList · React Native](https://reactnative.dev/docs/flatlist)

********

********

**Bài tập 4: Tạo một màn hình với các component đã học trên hiển thị 1 danh sách 100 item với các chuỗi tên random (có scrollview) như ví dụ trên**

## StyleSheet và Kiểu Style CSS

Trong React Native, bạn có thể sử dụng một số kiểu style để định dạng và tạo kiểu cho các thành phần giao diện của bạn. Dưới đây là một số kiểu style phổ biến:

****1. Inline Styles:****Bạn có thể sử dụng các thuộc tính trực tiếp trong các thành phần giao diện của bạn để áp dụng kiểu. Ví dụ:



2. External Styles with Stylesheet: Sử dụng `StyleSheet.create()` để định nghĩa các kiểu style trong một đối tượng riêng biệt và sau đó áp dụng chúng cho các thành phần. Điều này giúp tối ưu hóa hiệu suất vì kiểu style chỉ được tính toán một lần.

********

Thư Viện Hỗ Trợ (Tham khảo)

* React Native có một số thư viện hỗ trợ trong việc quản lý style và xây dựng các thành phần giao diện người dùng. Dưới đây là một số thư viện phổ biến mà bạn có thể sử dụng để cải thiện quá trình phát triển ứng dụng React Native của bạn:
* react-native-elements: Thư viện này cung cấp một bộ sưu tập các thành phần giao diện người dùng đẹp và có sẵn như nút, input, biểu đồ, avatar, và nhiều thành phần khác. Nó giúp bạn nhanh chóng xây dựng giao diện hấp dẫn mà không cần phải tạo từ đầu.
* NativeBase: NativeBase là một bộ thư viện chứa các thành phần UI cơ bản cho React Native. Nó cung cấp hơn 100+ thành phần và mẫu được tùy chỉnh dễ dàng, giúp bạn tạo ra giao diện đa dạng và hấp dẫn.
* React Native Paper: Thư viện này cung cấp các thành phần giao diện người dùng theo hướng Material Design. Nó giúp bạn tạo ra giao diện đẹp và hiện đại dễ dàng.
* React Native Vector Icons: Thư viện này cho phép bạn sử dụng biểu tượng vector từ nhiều bộ biểu tượng khác nhau trong ứng dụng của bạn. Điều này làm cho việc thêm biểu tượng vào giao diện trở nên dễ dàng hơn.
* react-native-gesture-handler: Thư viện này cung cấp các cơ chế cơ bản để xử lý cử chỉ và tương tác người dùng như vuốt, kéo, và nhấn giữ. Nó là một phần quan trọng của việc xây dựng các thành phần tương tác trong ứng dụng React Native.

Bài tập 5: Tạo một màn hình với các component cơ bản đã học như hình sau

1. Sử dụng component hiển thị hình ảnh (thay thế hình trong thẻ bằng hình ảnh bất kì)
2. Sử dụng component hiển thị thông tin cá nhân như hình dưới
3. Sử dụng component nút nhấn và tạo sự kiện in ra màn hình chữ ‘Hello React Native’
4. 