**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA KHOA HỌC VÀ KĨ THUẬT THÔNG TIN**



**BÁO CÁO ĐỒ ÁN GIỮA KỲ**

**MÔN CÔNG NGHỆ LẬP TRÌNH ĐA NỀN TẢNG**

**Đề tài: Tìm hiểu về Dropdown / Picker, Date Time Components, Calendar Components**

GVHD: ThS. Võ Ngọc Tân

Nhóm sinh viên thực hiện:

1. Phan Tấn Cảnh MSSV: 22540002

2. Nguyễn Tấn Đạt MSSV: 22540003

3. Huỳnh Trung Hiếu MSSV: 22540006

🙡🙢 Tp. Hồ Chí Minh, 11/2024 🙠🙣

**NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN**

*……., ngày……...tháng……năm 20…*

**Người nhận xét**

*(Ký tên và ghi rõ họ tên****)***

**MỤC LỤC**

[Phần 1: Dropdown/Picker 7](#_Toc181353804)

[Chương 1: Searchable Dropdown/Picker [1] 7](#_Toc181353805)

[1. Giới thiệu 7](#_Toc181353806)

[2. Cài đặt thư viện 7](#_Toc181353807)

[3. Sử dụng 7](#_Toc181353808)

[Chương 2: Multiple Select / Dropdown / Picker [2] 14](#_Toc181353809)

[1. Giới thiệu 14](#_Toc181353810)

[2. Cài đặt thư viện 14](#_Toc181353811)

[3. Sử dụng 14](#_Toc181353812)

[Phần 2: Date time components 19](#_Toc181353813)

[Chương 1: React Native Date Picker [3] 19](#_Toc181353814)

[1. Giới thiệu 19](#_Toc181353815)

[2. Cài đặt thư viện 19](#_Toc181353816)

[3. Sử dụng 19](#_Toc181353817)

[Chương 2: React Native Time Picker [4] 24](#_Toc181353818)

[1. Giới thiệu 24](#_Toc181353819)

[2. Cài đặt thư viện 24](#_Toc181353820)

[3. Sử dụng 24](#_Toc181353821)

[Chương 3: React Native Countdown Timer [5] 29](#_Toc181353822)

[1. Giới thiệu 29](#_Toc181353823)

[2. Cài đặt thư viện 29](#_Toc181353824)

[3. Sử dụng 29](#_Toc181353825)

[Chương 4: React Native Timer và React Native Stopwatch [6] 33](#_Toc181353826)

[1. Giới thiệu 33](#_Toc181353827)

[2. Cài đặt thư viện 33](#_Toc181353828)

[3. Sử dụng 33](#_Toc181353829)

[Phần 3: Calendar Components 39](#_Toc181353830)

[Chương 1: RN Calendar Picker [7] 39](#_Toc181353831)

[1. Giới thiệu 39](#_Toc181353832)

[2. Cách triển khai Calendar Picker 39](#_Toc181353833)

[Chương 2: Calendar with Events [8] 42](#_Toc181353834)

[1. Giới thiệu 42](#_Toc181353835)

[2. Cách triển khai 42](#_Toc181353836)

[Chương 3: Add Event in Device’s Calendar [9] 46](#_Toc181353837)

[1. Giới thiệu 46](#_Toc181353838)

[2. Cách triển khai 46](#_Toc181353839)

[Tài liệu tham khảo 54](#_Toc181353840)

**DANH MỤC HÌNH ẢNH**

[Hình 1: Kết quả sau khi cài đặt bộ chọn ngày sử dụng thư viện Searchable Dropdown/Picker 13](#_Toc181353855)

[Hình 2: Kết quả sau khi cài đặt bộ chọn ngày sử dụng thư viện react-native-mutil select, dropdown, picker 18](#_Toc181353856)

[Hình 3: Kết quả sau khi cài đặt bộ chọn ngày sử dụng thư viện react-native-datepicker 23](#_Toc181353857)

[Hình 4: Kết quả sau khi cài đặt bộ thời gian sử dụng thư viện react-native-simple-time-picker 28](#_Toc181353858)

[Hình 5: Kết quả sau khi cài đặt bộ đếm ngược thời gian sử dụng thư viện react-native-countdown-component 32](#_Toc181353859)

[Hình 6: Kết quả sau khi cài đặt bộ bấm giờ sử dụng thư viện react-native-stopwatch-timer 38](#_Toc181353860)

[Hình 7: Kết quả sau khi cài đặt và sử dụng thư viện react-native-calendar-picker 41](#_Toc181353861)

[Hình 8: Kết quả chạy ứng dụng sử dụng react-native-events-calendar 45](#_Toc181353862)

[Hình 9: Kết quả chạy ứng dụng với các sự kiện Add Event in Device’s Calendar 53](#_Toc181353863)

**DANH MỤC BẢNG**

[Bảng 1: Tóm tắt các thuộc tính Searchable dropdown picker 7](#_Toc181353966)

[Bảng 2: Mã nguồn cách sử dụng Searchabledropdown picker 9](#_Toc181353967)

[Bảng 3: Bảng tóm tắt cách thuộc tính Mutilple select, dropdown, Picker 14](#_Toc181353968)

[Bảng 4: Mã nguồn các sử dụng multiple select, dropdown,picker 15](#_Toc181353969)

[Bảng 5: Sự tương thích giữa các phiên bảng React Native và React Native Date Picker 19](#_Toc181353970)

[Bảng 6: Mã nguồn cách sử dụng React Native Date Picker 21](#_Toc181353971)

[Bảng 7: Tóm tắt các thuộc tính của một đối tượng react-native-simple-time-picker 24](#_Toc181353972)

[Bảng 8: Mã nguồn cách sử dụng React Native Time Picker 26](#_Toc181353973)

[Bảng 9: Tóm tắt các thuộc tính của một đối tượng react-native-countdown-component 29](#_Toc181353974)

[Bảng 10:Mã nguồn cách sử dụng React Native CountDown Component 30](#_Toc181353975)

[Bảng 11: Tóm tắt các thuộc tính của stopwatch và timer 33](#_Toc181353976)

[Bảng 12: Tóm tắt các thuộc tính của stopwatch 33](#_Toc181353977)

[Bảng 13: Tóm tắt các thuộc tính của timer 34](#_Toc181353978)

[Bảng 14: Mã nguồn cách sử dụng React Native Stopwatch and Timer 34](#_Toc181353979)

[Bảng 15: Mã nguồn cách sử dụng react-native-calendar-picker 40](#_Toc181353980)

[Bảng 16: Mã nguồn cấu hình EventCalendar 42](#_Toc181353981)

[Bảng 17: Mã nguồn cách sử dụng Calendar với Sự kiện 43](#_Toc181353982)

[Bảng 18: Cấu hình Add Event in Device’s Calendar 46](#_Toc181353983)

[Bảng 19: Mã nguồn để thêm sự kiện vào lịch thiết bị 47](#_Toc181353984)

# Dropdown/Picker

## Searchable Dropdown/Picker [1]

### Giới thiệu

*Searchable Dropdown/Picker* là một thành phần giao diện người dùng thường được sử dụng trong các ứng dụng di động và web để hỗ trợ người dùng tìm kiếm và chọn lựa từ một danh sách các mục. Khác với các thành phần dropdown truyền thống, Searchable Dropdown/Picker tích hợp thêm tính năng tìm kiếm cho phép người dùng lọc kết quả theo từ khóa, giúp cải thiện hiệu suất khi làm việc với danh sách lớn. Điều này đặc biệt hữu ích trong các ứng dụng có nhiều tùy chọn, khi việc cuộn qua toàn bộ danh sách trở nên không hiệu quả.

### Cài đặt thư viện

#### Yêu cầu

Để sử dụng *react-native-searchable-dropdown* trong dự án React Native, không có phiên bản tối thiểu chính thức của React Native được yêu cầu, tuy nhiên, vì đây là một thư viện đã ngừng bảo trì. Nếu sử dụng các phiên bản mới hơn của React Native (0.64 trở lên), có thể gặp vấn đề về tương thích, và cần phải điều chỉnh mã nguồn để phù hợp với môi trường của mình.

#### Cài đặt

Để cài đặt thư viện *react-native-searchable-dropdown* ta chạy câu lệnh sau :

npm install react-native-searchable-dropdown –save

### Sử dụng

#### Các thuộc tính

Bảng : Tóm tắt các thuộc tính Searchable dropdown picker

|  |  |  |  |
| --- | --- | --- | --- |
| **Thuộc tính** | **Kiểu dữ liệu** | **Giá trị mặc định** | **Mô tả** |
| items | Array of Objects | [] | Mảng các mục để hiển thị trong dropdown. Mỗi mục là một đối tượng. {id: number, name: string}. |
| defaultIndex | Number | -1 | Vị trí của mục được chọn mặc định trong danh sách. |
| onTextChange | function | undefined | Hàm được gọi khi văn bản trong ô tìm kiếm thay đổi. |
| onItemSelect | Function | undefined | Hàm được gọi khi người dùng chọn một mục từ danh sách. |
| ContainerStyle | Object | {} | Kiểu CSS cho khung chứa toàn bộ dropdown |
| textInputStyle | Object | {} | Kiểu CSS cho ô nhập liệu, bao gồm thuộc tính viền và nền. |
| itemStyle | Object | {} | Kiểu CSS cho từng mục trong danh sách. |
| itemsContainerStyle | Object | {} | Kiểu CSS cho vùng chứa các mục, có thể giới hạn chiều cao của danh sách qua maxHeight. |
| resetValue | Boolean | false | Đặt lại giá trị của ô tìm kiếm khi người dùng chọn một mục. |
| underlineColorAndroid | String | "transparent" | Loại bỏ đường gạch chân mặc định trên Android |
| placeholder | String | “” | Văn bản hiển thị khi ô tìm kiếm trống |
| listProps | Object | {} | Các thuộc tính của FlatList, như nestedScrollEnabled. |
| textInputProps | Object | {} | Các thuộc tính của TextInput, như màu placeholder và kiểu đường viền. |
| multi | Boolean | false | Cho phép chọn nhiều mục. |
| selectedItems | Array of Objects | [] | Mảng các mục đã chọn (chỉ dùng khi multi là true). |
| chip | Boolean | false | Hiển thị các mục đã chọn dưới dạng chip (chỉ dùng với multi). |
| onRemoveItem | Function | undefined | Hàm gọi khi xóa một mục đã chọn trong chế độ multi. |

#### Ví dụ sử dụng

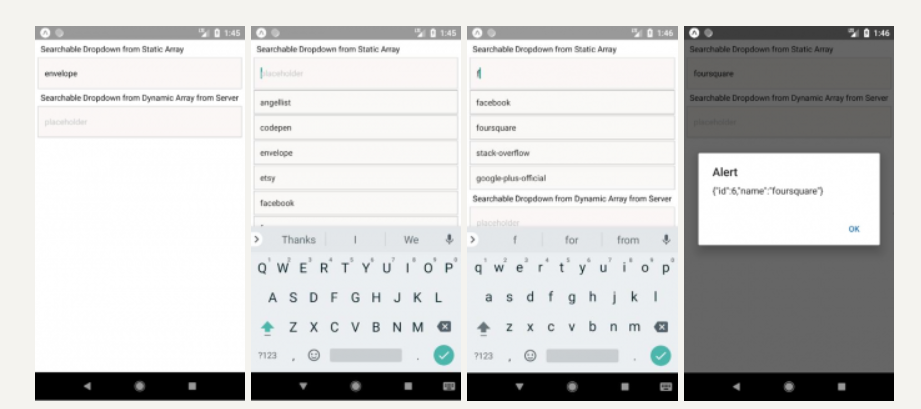
Trong React Native, Searchable Dropdown được hỗ trợ bởi các thư viện như react-native-searchable-dropdown. Thành phần này cho phép người phát triển cấu hình linh hoạt, từ kiểu dáng giao diện người dùng đến các sự kiện và hàm callback nhằm kiểm soát hành vi tương tác của người dùng, ví dụ: onTextChange để lắng nghe thay đổi từ khóa tìm kiếm, và onItemSelect để xử lý lựa chọn của người dùng. Thông qua các hàm callback này, ứng dụng có thể xử lý dữ liệu một cách hiệu quả, đồng thời cung cấp phản hồi tức thì cho người dùng khi chọn các mục trong danh sách.

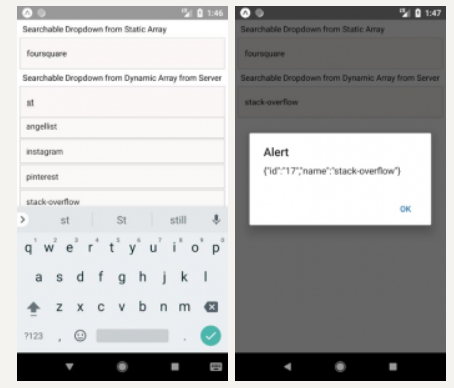
Một trong những lợi ích đáng kể của Searchable Dropdown/Picker là khả năng tích hợp với dữ liệu động từ các nguồn bên ngoài như API. Điều này cho phép ứng dụng linh hoạt trong việc cập nhật và hiển thị dữ liệu theo thời gian thực, nâng cao trải nghiệm người dùng khi tương tác với dữ liệu thay đổi thường xuyên. Dưới đây là mã nguồn chi tiết cho ứng dụng này.

Bảng : Mã nguồn cách sử dụng Searchabledropdown picker

|  |
| --- |
| import React, {useState, useEffect} from 'react';  import {SafeAreaView, StyleSheet, Text, View} from 'react-native';  import SearchableDropdown from 'react-native-searchable-dropdown';  const items = [  {id: 1, name: 'angellist'},  {id: 2, name: 'codepen'},  {id: 3, name: 'envelope'},  {id: 4, name: 'etsy'},  {id: 5, name: 'facebook'},  {id: 6, name: 'foursquare'},  {id: 7, name: 'github-alt'},  {id: 8, name: 'github'},  {id: 9, name: 'gitlab'},  {id: 10, name: 'instagram'},  ];  const App = () => {  const [serverData, setServerData] = useState([]);  useEffect(() => {  fetch('https://abooutreactapis.000webhostapp.com/demosearchables.php')  .then((response) => response.json())  .then((responseJson) => setServerData(responseJson.results))  .catch((error) => console.error(error));  }, []);  return (  <SafeAreaView style={styles.container}>  <View style={styles.container}>  <Text style={styles.titleText}>  Ví dụ về Dropdown / Picker có khả năng tìm kiếm trong React Native  </Text>  <Text style={styles.headingText}>  Dropdown có thể tìm kiếm từ mảng tĩnh  </Text>  <SearchableDropdown  onTextChange={(text) => console.log(text)}  onItemSelect={(item) => alert(JSON.stringify(item))}  containerStyle={{padding: 5}}  textInputStyle={{  padding: 12,  borderWidth: 1,  borderColor: '#ccc',  backgroundColor: '#FAF7F6',  }}  itemStyle={{  padding: 10,  marginTop: 2,  backgroundColor: '#FAF9F8',  borderColor: '#bbb',  borderWidth: 1,  }}  itemTextStyle={{ color: '#222' }}  itemsContainerStyle={{ maxHeight: '60%' }}  items={items}  defaultIndex={2}  placeholder="placeholder"  resetValue={false}  underlineColorAndroid="transparent"  />  <Text style={styles.headingText}>  Dropdown có thể tìm kiếm từ mảng động từ server  </Text>  <SearchableDropdown  onTextChange={(text) => console.log(text)}  onItemSelect={(item) => alert(JSON.stringify(item))}  containerStyle={{padding: 5}}  textInputStyle={{  padding: 12,  borderWidth: 1,  borderColor: '#ccc',  backgroundColor: '#FAF7F6',  }}  itemStyle={{  padding: 10,  marginTop: 2,  backgroundColor: '#FAF9F8',  borderColor: '#bbb',  borderWidth: 1,  }}  itemTextStyle={{ color: '#222' }}  itemsContainerStyle={{ maxHeight: '50%' }}  items={serverData}  defaultIndex={2}  placeholder="placeholder"  resetValue={false}  underlineColorAndroid="transparent"  />  </View>  </SafeAreaView>);};  export default App;  const styles = StyleSheet.create({  container: {  flex: 1,  backgroundColor: 'white',  padding: 10,  },  titleText: {  padding: 8,  fontSize: 16,  textAlign: 'center',  fontWeight: 'bold',  },  headingText: {  padding: 8,  },  }); |

*Kết quả :*





Hình : Kết quả sau khi cài đặt bộ chọn ngày sử dụng thư viện Searchable Dropdown/Picker

## Multiple Select / Dropdown / Picker [2]

### Giới thiệu

Multiple Select, Picker, hoặc Dropdown là các thành phần giao diện người dùng (UI) quan trọng trong phát triển ứng dụng di động và web, cho phép người dùng chọn một hoặc nhiều mục từ danh sách các tùy chọn có sẵn. Chức năng này thường được áp dụng trong các tình huống mà người dùng cần lựa chọn nhiều mục liên quan từ tập hợp lớn hoặc khi việc liệt kê trực tiếp mọi tùy chọn không hiệu quả về mặt không gian hiển thị.

### Cài đặt thư viện

#### Yêu cầu

Phiên bản của react-native-multiple-select yêu cầu sử dụng React Native từ phiên bản 0.60 trở lên. Điều này đảm bảo rằng bạn có hỗ trợ đầy đủ cho các tính năng mà thư viện cung cấp, như cấu hình CocoaPods cho iOS và các API hiện đại.

#### Cài đặt

Để cài đặt thư biện Multi Select, Dropdown, Picker ta chạy câu lệnh sau :

npm install react-native-multiple-select –save

**Cài đặt thêm react-native-vector-icons** để có thể sử dụng biểu tượng chọn mục:

npm install react-native-vector-icons --save

### Sử dụng

#### Các thuộc tính

Bảng : Bảng tóm tắt cách thuộc tính Mutilple select, dropdown, Picker

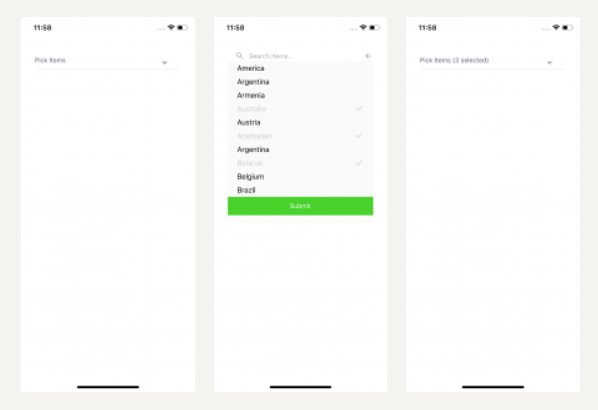
|  |  |  |  |
| --- | --- | --- | --- |
| **Thuộc tính** | **Kiểu dữ liệu** | **Giá trị mặc định** | **Mô tả** |
| items | Array | [] | Mảng các mục để hiển thị trong dropdown. Mỗi mục là một đối tượng. {id: number, name: string}. |
| uniqueKey | String | Null | Khóa duy nhất để xác định mỗi mục, thường là ID. |
| selectedItems | Array | [] | Mảng các mục đã chọn, chứa các id của mục. |
| onSelectedItemsChange | Function | Null | Hàm được gọi khi có thay đổi trong các mục đã chọn, nhận tham số là mảng selectedItems. |
| selectText | String | "Select..." | Văn bản hiển thị khi chưa có mục nào được chọn. |
| SearchInputPlaceholderText | String | "Search..." | Văn bản hiển thị trên thanh tìm kiếm. |
| onChangeInput | Function | Null | Hàm được gọi khi văn bản trong ô tìm kiếm thay đổi. |
| tagRemoveIconColor | String (hex) | "#CCC" | Màu của biểu tượng xóa bên cạnh thẻ mục đã chọn. |
| tagBorderColor | String (hex) | "#CCC" | Màu viền của thẻ mục đã chọn. |
| tagTextColor | String (hex) | "#CCC" | Màu văn bản của mục được chọn trong danh sách. |
| selectedItemIconColor | String (hex) | "#CCC" | Màu của biểu tượng mục đã chọn trong danh sách. |
| itemTextColor | String (hex) | "#000" | Màu văn bản của các mục trong danh sách. |
| displayKey | String | "name" | Khóa để hiển thị văn bản của mục (ví dụ: name). |
| searchInputStyle | Object | false | Định dạng cho trường tìm kiếm, như color |
| submitButtonColor | String (hex) | "#48d22b" | Màu của nút xác nhận chọn. |
| submitButtonText | String | "Submit" | Văn bản trên nút xác nhận chọn. |
| hideTags | Boolean | false | Ẩn các thẻ mục đã chọn nếu true. |

#### Ví dụ sử dụng

Bảng : Mã nguồn các sử dụng multiple select, dropdown,picker

|  |
| --- |
| // Multiple Select / Dropdown / Picker Example in React Native  // https://aboutreact.com/multiple-select-dropdown-picker-example-in-react-native/  // import React in our code  import React, {useState, useEffect} from 'react';  // import all the components we are going to use  import {SafeAreaView, StyleSheet, Text, View} from 'react-native';  // import MultiSelect library  import MultiSelect from 'react-native-multiple-select';  // Dummy Data for the MutiSelect  const items = [  // name key is must. It is to show the text in front  {id: 1, name: 'angellist'},  {id: 2, name: 'codepen'},  {id: 3, name: 'envelope'},  {id: 4, name: 'etsy'},  {id: 5, name: 'facebook'},  {id: 6, name: 'foursquare'},  {id: 7, name: 'github-alt'},  {id: 8, name: 'github'},  {id: 9, name: 'gitlab'},  {id: 10, name: 'instagram'},  ];  const App = () => {  // Data Source for the SearchableDropdown  const [selectedItems, setSelectedItems] = useState([]);  const onSelectedItemsChange = (selectedItems) => {  // Set Selected Items  setSelectedItems(selectedItems);  };  useEffect(() => {  fetch('https://abooutreactapis.000webhostapp.com/demosearchables.php')  .then((response) => response.json())  .then((responseJson) => {  //Successful response from the API Call  setServerData(responseJson.results);  })  .catch((error) => {  console.error(error);  });  }, []);  return (  <SafeAreaView style={styles.container}>  <View style={styles.container}>  <Text style={styles.titleText}>  Multiple Select / Dropdown / Picker Example  in React Native  </Text>  <MultiSelect  hideTags  items={items}  uniqueKey="id"  onSelectedItemsChange={onSelectedItemsChange}  selectedItems={selectedItems}  selectText="Pick Items"  searchInputPlaceholderText="Search Items..."  onChangeInput={(text) => console.log(text)}  tagRemoveIconColor="#CCC"  tagBorderColor="#CCC"  tagTextColor="#CCC"  selectedItemTextColor="#CCC"  selectedItemIconColor="#CCC"  itemTextColor="#000"  displayKey="name"  searchInputStyle={{color: '#CCC'}}  submitButtonColor="#48d22b"  submitButtonText="Submit"  />  </View>  </SafeAreaView>  );  };  export default App;  const styles = StyleSheet.create({  container: {  flex: 1,  backgroundColor: 'white',  padding: 10,  },  titleText: {  padding: 8,  fontSize: 16,  textAlign: 'center',  fontWeight: 'bold',  },  headingText: {  padding: 8,  },  }); |

*Kết quả :*



Hình : Kết quả sau khi cài đặt bộ chọn ngày sử dụng thư viện react-native-mutil select, dropdown, picker

# Date time components

## React Native Date Picker [3]

### Giới thiệu

*React Native Date Picker* là một thành phần giúp người dùng chọn ngày và giờ trong ứng dụng di động React Native. Việc chọn ngày giờ là một yếu tố thiết yếu trong các ứng dụng liên quan đến lịch trình, sự kiện, hoặc bất kỳ nơi nào cần quản lý thời gian. Thành phần này mang lại trải nghiệm người dùng thân thiện và hiệu quả trong việc chọn ngày giờ, đặc biệt khi phát triển ứng dụng di động đa nền tảng với React Native.

### Cài đặt thư viện

#### Yêu cầu

Phiên bản React Native tối thiểu hỗ trợ cho thư viện *React Native Date Picker* là >= 0.57, chi tiết các phiên bản tương thích khác được trình bài ở (bảng 5).

Bảng : Sự tương thích giữa các phiên bảng React Native và React Native Date Picker

|  |  |
| --- | --- |
| **Phiên bản React Native** | **Phiên bản React Native Date Picker** |
| >= 0.64 | >=4.0.0 |
| 0.60 – 0.63 | 3.0.0 – 4.0.0 |
| < 0.60 | <= 2.0.0 |

#### Cài đặt

Để cài đặt thư viện React Native Date Picker chạy câu lệnh sau:

npm install react-native-datepicker –save

### Sử dụng

#### Các thuộc tính

*React Native Date Picker* cung cấp các thuộc tính để hỗ trợ người dùng cài đặt và sử dụng một cách hiệu quả. Dưới đây là danh sách các thuộc tính và chức năng của chúng:

* *date*: ngày hiện tại được chọn.
* *onDateChange*: bộ xử lý sự kiện thay đổi ngày (chỉ dùng khi ở chế độ *inline*)*.*
* *maximumDate*: ngày tối đa có thể chọn. Ví dụ: new Date("2021-12-31").
* *minimunDate*: ngày tối thiểu có thể chọn. Ví dụ: new Date(“2021-01-01”).
* *minuteInterval*: khoảng thời gian giữa các phút có thể chọn.
* *mode*: chế độ của Date Picker, có thể là "datetime", "date", hoặc "time".
* *locale*: ngôn ngữ của Date Picker, thay đổi thứ tự ngày, tháng, năm và chế độ AM/PM tùy theo locale.
* *timeZoneOffsetInMinutes*: sự lệch múi giờ tính bằng phút (mặc định theo múi giờ của thiết bị).
* *is24hourSource*: xác định chế độ 24 giờ (AM/PM) dựa trên cài đặt của thiết bị hoặc ngôn ngữ. Các giá trị có thể là 'locale', 'device' (chỉ hỗ trợ Android, mặc định: 'device').
* *modal*: giá trị boolean xác định nếu Date Picker sẽ hiển thị dạng modal. Mặc định là "false". Khi bật, cần sử dụng các thuộc tính modal khác.
* *theme*: chủ đề của modal. Có thể là "light", "dark", hoặc "auto". Mặc định là "auto".
* *open*: (chỉ dùng cho modal) giá trị boolean xác định nếu modal sẽ mở.
* *onConfirm*: (chỉ dùng cho modal) Callback trả về ngày khi người dùng bấm vào nút xác nhận.
* *onCancel*: (chỉ dùng cho modal) Callback khi người dùng bấm hủy hoặc đóng modal bằng cách bấm bên ngoài modal.
* *title*: (chỉ dùng cho modal) Văn bản tiêu đề. Có thể đặt là null để loại bỏ văn bản.
* *confirmText*: (chỉ dùng cho modal) Văn bản của nút xác nhận.
* *cancelText*: (chỉ dùng cho modal) Văn bản của nút hủy.
* *buttonColor*: màu sắc của nút xác nhận và nút hủy. (chỉ dành cho modal trên Android).
* *dividerColor*: màu sắc của dải phân cách / đường ngăn. (chỉ hỗ trợ Android).
* *onStateChange*: bộ xử lý sự kiện thay đổi trạng thái spinner, kích hoạt khi chuyển giữa trạng thái "idle" và "spinning" (chỉ hỗ trợ Inline trên Android).

#### Ví dụ sử dụng

*Cài đặt mã nguồn:*

(Bảng 6) trình bày cách triển khai bộ chọn ngày sử dụng thư viện *react-native-datepicker* trong ứng dụng React Native. Mã nguồn này không chỉ cung cấp chức năng chọn ngày mà còn hỗ trợ tùy chỉnh giao diện theo nhu cầu của người phát triển. Cụ thể, mã nguồn cho phép người dùng chọn một ngày trong khoảng từ 01-01-2016 đến 01-01-2019, với định dạng ngày là DD-MM-YYYY. Ngoài ra, nó cũng bao gồm các nút xác nhận và hủy, mang lại sự linh hoạt cho người dùng trong quá trình chọn ngày. Việc cài đặt mã nguồn này sẽ giúp lập trình viên dễ dàng tích hợp bộ chọn ngày vào ứng dụng, từ đó nâng cao tính tương tác và trải nghiệm của người dùng. Dưới đây là mã nguồn chi tiết cho ứng dụng này.

Bảng : Mã nguồn cách sử dụng React Native Date Picker

|  |
| --- |
| import React, { useState } from 'react';  import { SafeAreaView, StyleSheet, Text, View } from 'react-native';  import DatePicker from 'react-native-datepicker';  const App = () => {  const [date, setDate] = useState('09-10-2020');  return (  <SafeAreaView style={styles.container}>  <View style={styles.container}>  <Text style={styles.title}>  Example of Collapsible/Accordion/Expandable List View in  React Native  </Text>  <DatePicker  style={styles.datePickerStyle}  date={date}  mode="date"  placeholder="select date"  format="DD-MM-YYYY"  minDate="01-01-2016"  maxDate="01-01-2019"  confirmBtnText="Confirm"  cancelBtnText="Cancel"  customStyles={{  dateIcon: {  //display: 'none',  position: 'absolute',  left: 0,  top: 4,  marginLeft: 0,  },  dateInput: {  marginLeft: 36,  },  }}  onDateChange={(date) => {  setDate(date);  }}  />  </View>  </SafeAreaView>  );  };  export default App;  const styles = StyleSheet.create({  container: {  flex: 1,  padding: 10,  justifyContent: 'center',  alignItems: 'center',  },  title: {  textAlign: 'center',  fontSize: 20,  fontWeight: 'bold',  padding: 20,  },  datePickerStyle: {  width: 200,  marginTop: 20,  },  }); |

*Kết quả*

Screens screenshot of a calendar

Description automatically generated

Hình 3: Kết quả sau khi cài đặt bộ chọn ngày sử dụng thư viện react-native-datepicker

## React Native Time Picker [4]

### Giới thiệu

Trong quá trình phát triển ứng dụng di động, việc cung cấp cho người dùng khả năng chọn thời gian một cách dễ dàng và trực quan là rất quan trọng. Thư *viện react-native-simple-time-picker* được thiết kế để đơn giản hóa quá trình này, cho phép người dùng lựa chọn thời gian một cách nhanh chóng và thuận tiện. Thư viện này cung cấp một giao diện thân thiện và dễ sử dụng, phù hợp cho nhiều loại ứng dụng từ quản lý thời gian đến đặt lịch hẹn.

### Cài đặt thư viện

#### Yêu cầu

Để sử dụng thư viện react-native-simple-time-picker, cần đảm bảo rằng dự án đã cài đặt các phiên bản tương thích của các thư viện liên quan. Cụ thể, yêu cầu phiên bản cho thư viện này là:

* React Native: Phiên bản từ 0.60 trở lên
* *react-native-simple-time-picker*: Phiên bản mới nhất (có thể được cài đặt thông qua npm hoặc yarn)

#### Cài đặt

Để cài đặt thư viện *react-native-simple-time-picker*, chạy câu lệnh sau:

npm install react-native-simple-time-picker –save

### Sử dụng

#### Các thuộc tính

Để sử dụng một cách hiệu quả và thuận tiện, *react-native-simple-time-picker* cung cấp cho người dùng một loạt các thuộc tính để thao tác trên một đối tượng được tạo ra từ thư viện này (Bảng 7).

Bảng : Tóm tắt các thuộc tính của một đối tượng react-native-simple-time-picker

|  |  |  |  |
| --- | --- | --- | --- |
| **Thuộc tính** | **Kiểu dữ liệu** | **Giá trị mặc định** | **Mô tả** |
| value | { hours: number, minutes: number, seconds: number, ampm?: 'am' | 'pm' } | { hours: 0, minutes: 0, seconds: 0 } | Giá trị thời gian hiện tại được chọn. |
| defaultValue | { hours: number, minutes: number, seconds: number, ampm?: 'am' | 'pm' } | { hours: 0, minutes: 0, seconds: 0 } | Giá trị thời gian mặc định. |
| onChange | function |  | Callback khi value thay đổi. |
| pickerShows | Array | ["hours", "minutes"] | Mảng xác định các picker (bộ chọn) sẽ được hiển thị. Ví dụ: ["hours", "minutes", "seconds"] để hiển thị cả ba. |
| hoursUnit | String | “” | Đơn vị của giờ cho nhãn. Ví dụ: "giờ". |
| minutesUnit | String | '' | Đơn vị của phút cho nhãn. Ví dụ: "phút". |
| secondsUnit | String | '' | Đơn vị của giây cho nhãn. Ví dụ: "giây". |
| zeroPadding | Boolean | false | Chỉ định xem có nên thêm số 0 vào các nhãn số hay không (ví dụ: hiển thị 01 thay vì 1). |
| textColor | String |  | Chỉ định màu sắc của văn bản đối tượng thời gian được chọn. |
| hoursInterval | Integer | 1 | Khoảng cách giữa các giá trị giờ trong picker. Ví dụ: 1 có nghĩa là người dùng có thể chọn từng giờ một. |
| minutesInterval | Integer | 1 | Khoảng cách giữa các giá trị phút trong picker. Ví dụ: 5 cho phép chọn phút cách nhau 5 phút (0, 5, 10, 15,...). |
| secondsInterval | Integer | 1 | Khoảng cách giữa các giá trị giây trong picker. Tương tự như minutesInterval. |
| emptyLabel | String | undefined | Bật tùy chọn rỗng với nhãn này. Nếu được cung cấp, người dùng có thể chọn tùy chọn rỗng (không chọn thời gian). |
| isAmpm | Boolean | false | Xác định xem có hiển thị picker am/pm hay không. |
| ampmLocalization | { am: string, pm: string } | { am: 'am', pm: 'pm' } | Nhãn cho các mục picker am/pm, cho phép tùy chỉnh cách hiển thị (ví dụ: sử dụng "sáng" và "tối" thay vì "am" và "pm"). |

#### Ví dụ sử dụng

*Cài đặt mã nguồn*

(Bảng 8) mô tả cài đặt một ứng dụng sử dụng bộ chọn thời gian (TimePicker) được nhập từ thư *viện react-native-simple-time-picker*, cho phép người dùng tương tác và chọn thời gian một cách trực quan. Với việc sử dụng thư viện *react-native-simple-time-picker*, ứng dụng đảm bảo tính hiệu quả và trải nghiệm người dùng tốt, đồng thời cho phép tích hợp dễ dàng vào các ứng dụng React Native khác.

Bảng : Mã nguồn cách sử dụng React Native Time Picker

|  |
| --- |
| import React, { useState } from 'react';  import { SafeAreaView, StyleSheet, Text, View } from 'react-native';  import TimePicker from 'react-native-simple-time-picker';  const App = () => {  const [selectedHours, setSelectedHours] = useState(0);  const [selectedMinutes, setSelectedMinutes] = useState(0);  return (  <SafeAreaView style={styles.container}>  <View style={styles.container}>  <Text style={styles.title}>  React Native Time Picker – To Pick the Time using Native  Time Picker  </Text>  <Text>  Selected Time: {selectedHours}:{selectedMinutes}  </Text>  <TimePicker  selectedHours={selectedHours}  //initial Hourse value  selectedMinutes={selectedMinutes}  //initial Minutes value  onChange={(hours, minutes) => {  setSelectedHours(hours);  setSelectedMinutes(minutes);  }}  />  </View>  </SafeAreaView>  );  };  export default App;  const styles = StyleSheet.create({  container: {  flex: 1,  padding: 10,  justifyContent: 'center',  alignItems: 'center',  },  title: {  textAlign: 'center',  fontSize: 20,  fontWeight: 'bold',  padding: 20,  },  }); |

*Kết quả*

A screenshot of a phone

Description automatically generated

Hình 4: Kết quả sau khi cài đặt bộ thời gian sử dụng thư viện react-native-simple-time-picker

## React Native Countdown Timer [5]

### Giới thiệu

Trong ứng dụng di động, bộ đếm ngược (countdown timer) là thành phần cho phép người dùng theo dõi thời gian còn lại cho một sự kiện hoặc hoạt động cụ thể. Thư viện *react-native-countdown-component* là một giải pháp hiệu quả và dễ dàng để tích hợp tính năng đếm ngược vào ứng dụng React Native. Với cú pháp đơn giản và tính năng tùy biến cao, thư viện này giúp lập trình viên nhanh chóng triển khai các bộ đếm thời gian với giao diện bắt mắt và chức năng mạnh mẽ.

### Cài đặt thư viện

Để cài đặt thư viện *react-native-countdown-component*, chạy câu lệnh sau:

npm install react-native-countdown-component –save

### Sử dụng

#### Các thuộc tính

*react-native-countdown-component* cung cấp các thuộc tính hỗ trợ người dùng sử dụng một cách hiệu quả. (Bảng) mô tả các thuộc tính của *react-native-countdown-component.*

Bảng : Tóm tắt các thuộc tính của một đối tượng react-native-countdown-component

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên thuộc tính** | **Mô tả** | **Kiểu dữ liệu** | **Giá trị mặc định** |
| id | ID của bộ đếm, để xác định xem có nên reset bộ đếm hay không | string | null |
| style | Ghi đè kiểu của thành phần | object | {} |
| digitStyle | Kiểu của chữ số | object | {backgroundColor: '#FAB913'} |
| digitTxtStyle | Kiểu chữ của chữ số | object | {color: '#000'} |
| timeLabelStyle | Kiểu của nhãn thời gian | object | {color: '#000'} |
| separatorStyle | Kiểu của dấu phân cách | object | {color: '#000'} |
| size | Kích thước của thành phần đếm ngược | number | 15 |
| until | Số giây để đếm ngược | number | 0 |
| onFinish | Hàm nào sẽ được gọi khi thời gian còn lại là 0 | func | null |
| onChange | Hàm nào sẽ được gọi khi bộ đếm đang thay đổi | func | null |
| onPress | Hàm nào sẽ được gọi khi nhấp vào bộ đếm | func | null |
| timeToShow | Các chữ số nào sẽ hiển thị | array | ['D', 'H', 'M', 'S'] |
| timeLabels | Văn bản hiển thị trong nhãn thời gian | object | {d: 'Days', h: 'Hours', m: 'Minutes', s: 'Seconds'} |
| showSeparator | Có nên hiển thị dấu phân cách hay không | bool | FALSE |
| running | Một biến boolean để tạm dừng và tiếp tục thành phần | bool | TRUE |

#### Ví dụ sử dụng

*Cài đặt mã nguồn*

Dưới đây là ví dụ về ứng dụng react-native-countdown-component để xây dựng một ứng dụng count down (bảng 10).

Bảng :Mã nguồn cách sử dụng React Native CountDown Component

|  |
| --- |
| import React, { useState, useEffect } from 'react';  import { SafeAreaView, StyleSheet, Text, View } from 'react-native';  import CountDown from 'react-native-countdown-component';  import moment from 'moment';  const App = () => {  const [totalDuration, setTotalDuration] = useState(0);  useEffect(() => {  var date = moment().utcOffset('+05:30').format('YYYY-MM-DD hh:mm:ss');  var expirydate = '2020-12-23 04:00:45';  var diffr = moment.duration(moment(expirydate).diff(moment(date)));  var hours = parseInt(diffr.asHours());  var minutes = parseInt(diffr.minutes());  var seconds = parseInt(diffr.seconds());  var d = hours \* 60 \* 60 + minutes \* 60 + seconds;  setTotalDuration(d);  }, []);  return (  <SafeAreaView style={styles.container}>  <View style={styles.container}>  <Text style={styles.title}>  React Native CountDown Timer |  react-native-countdown-component  </Text>  <CountDown  until={totalDuration}  timetoShow={('H', 'M', 'S')}  onFinish={() => alert('finished')}  onPress={() => alert('hello')}  size={20}  />  </View>  </SafeAreaView>  );  };  export default App;  const styles = StyleSheet.create({  container: {  flex: 1,  padding: 10,  justifyContent: 'center',  alignItems: 'center',  },  title: {  textAlign: 'center',  fontSize: 20,  fontWeight: 'bold',  padding: 20,  },  }); |

*Kết quả*

A screenshot of a clock

Description automatically generated

Hình 5: Kết quả sau khi cài đặt bộ đếm ngược thời gian sử dụng thư viện react-native-countdown-component

## React Native Timer và React Native Stopwatch [6]

### Giới thiệu

Thư viện *react-native-stopwatch-timer* cung cấp một công cụ đồng hồ bấm giờ và hẹn giờ (timer) tiện lợi và dễ sử dụng cho các ứng dụng React Native. Với các thành phần tùy biến và linh hoạt, thư viện này cho phép phát triển các ứng dụng đếm thời gian chính xác và trực quan, thích hợp cho nhiều mục đích như thể dục, quản lý thời gian cá nhân, và các hoạt động đo lường thời gian thực.

### Cài đặt thư viện

Để cài đặt thư viện *react-native-stopwatch-timer*, sử dụng câu lệnh sau:

npm install react-native-stopwatch-timer --save

### Sử dụng

#### Các thuộc tính

(Bảng 11, 12, 13) mô tả các thuộc tính của các đối tượng của *react-native-stopwatch-timer*

Bảng : Tóm tắt các thuộc tính của stopwatch và timer

|  |  |  |  |
| --- | --- | --- | --- |
| **Thuộc tính** | **Kiểu dữ liệu** | **Mô tả** | **Giá trị mặc định** |
| start | boolean | Bắt đầu đồng hồ bấm giờ/timer nếu giá trị là true, dừng nếu là false | FALSE |
| reset | boolean | Dừng đồng hồ bấm giờ/timer và thiết lập lại | FALSE |
| msecs | boolean | Hiển thị mili giây trong thời gian hiển thị | FALSE |
| options | object | Mô tả kiểu hiển thị của đồng hồ bấm giờ/timer | Xem ví dụ |
| getTime | function | Hàm callback để nhận giá trị thời gian định dạng mỗi lần tick | (time) => console.log(time) |
| getMsecs | function | Hàm callback để nhận giá trị mili giây mỗi lần tick | (time) => console.log(time) |

Bảng : Tóm tắt các thuộc tính của stopwatch

|  |  |  |  |
| --- | --- | --- | --- |
| **Thuộc tính** | **Kiểu dữ liệu** | **Mô tả** | **Giá trị mặc định** |
| laps | boolean | Không đếm số vòng của đồng hồ bấm giờ khi dừng | FALSE |
| startTime | number | Số mili giây để bắt đầu đồng hồ bấm giờ từ | 0 |

Bảng : Tóm tắt các thuộc tính của timer

|  |  |  |  |
| --- | --- | --- | --- |
| **Thuộc tính** | **Kiểu dữ liệu** | **Mô tả** | **Giá trị mặc định** |
| totalDuration | Integer | Số mili giây để đặt thời gian cho bộ đếm | 0 |
| handleFinish | function | Hàm thực hiện khi bộ đếm thời gian hoàn tất | () => alert("Timer Finished") |

#### Ví dụ sử dụng

*Cài đặt mã nguồn*

Dưới đây là ví dụ về ứng dụng *react-native-stopwatch-timer* để xây dựng một ứng dụng bấm giờ (bảng 14).

Bảng : Mã nguồn cách sử dụng React Native Stopwatch and Timer

|  |
| --- |
| import React, { useState } from 'react';  import {  SafeAreaView,  StyleSheet,  Text,  View,  TouchableHighlight,  } from 'react-native';  import { Stopwatch, Timer } from 'react-native-stopwatch-timer';  const App = () => {  const [isTimerStart, setIsTimerStart] = useState(false);  const [isStopwatchStart, setIsStopwatchStart] = useState(false);  const [timerDuration, setTimerDuration] = useState(90000);  const [resetTimer, setResetTimer] = useState(false);  const [resetStopwatch, setResetStopwatch] = useState(false);  return (  <SafeAreaView style={styles.container}>  <View style={styles.container}>  <Text style={styles.title}>  Example of React Native Timer and Stopwatch  </Text>  <View style={styles.sectionStyle}>  <Stopwatch  laps  msecs  start={isStopwatchStart}  //To start  reset={resetStopwatch}  //To reset  options={options}  //options for the styling  getTime={(time) => {  console.log(time);  }}  />  <TouchableHighlight  onPress={() => {  setIsStopwatchStart(!isStopwatchStart);  setResetStopwatch(false);  }}  >  <Text style={styles.buttonText}>  {!isStopwatchStart ? 'START' : 'STOP'}  </Text>  </TouchableHighlight>  <TouchableHighlight  onPress={() => {  setIsStopwatchStart(false);  setResetStopwatch(true);  }}  >  <Text style={styles.buttonText}>RESET</Text>  </TouchableHighlight>  </View>  <View style={styles.sectionStyle}>  <Timer  totalDuration={timerDuration}  msecs  //Time Duration  start={isTimerStart}  //To start  reset={resetTimer}  //To reset  options={options}  //options for the styling  handleFinish={() => {  alert('Custom Completion Function');  }}  //can call a function On finish of the time  getTime={(time) => {  console.log(time);  }}  />  <TouchableHighlight  onPress={() => {  setIsTimerStart(!isTimerStart);  setResetTimer(false);  }}  >  <Text style={styles.buttonText}>  {!isTimerStart ? 'START' : 'STOP'}  </Text>  </TouchableHighlight>  <TouchableHighlight  onPress={() => {  setIsTimerStart(false);  setResetTimer(true);  }}  >  <Text style={styles.buttonText}>RESET</Text>  </TouchableHighlight>  </View>  </View>  </SafeAreaView>  );  };  export default App;  const styles = StyleSheet.create({  container: {  flex: 1,  padding: 10,  justifyContent: 'center',  alignItems: 'center',  },  title: {  textAlign: 'center',  fontSize: 20,  fontWeight: 'bold',  padding: 20,  },  sectionStyle: {  flex: 1,  marginTop: 32,  alignItems: 'center',  justifyContent: 'center',  },  buttonText: {  fontSize: 20,  marginTop: 10,  },  });  const options = {  container: {  backgroundColor: '#FF0000',  padding: 5,  borderRadius: 5,  width: 200,  alignItems: 'center',  },  text: {  fontSize: 25,  color: '#FFF',  marginLeft: 7,  },  }; |

*Kết quả*

A screenshot of a phone

Description automatically generated

Hình 6: Kết quả sau khi cài đặt bộ bấm giờ sử dụng thư viện react-native-stopwatch-timer

# Calendar Components

## RN Calendar Picker [7]

### Giới thiệu

Calendar Components giúp cung cấp khả năng lựa chọn và quản lý ngày tháng, rất hữu ích cho các ứng dụng yêu cầu tính năng này như ứng dụng quản lý công việc, lịch trình cá nhân, v.v. Các thư viện như react-native-calendar-picker cung cấp các công cụ sẵn sàng để dễ dàng tích hợp vào ứng dụng, hỗ trợ cả trên Android và iOS.

React-native-calendar-picker là một thư viện phổ biến trong React Native giúp tạo một Calendar Picker, cho phép người dùng chọn ngày hoặc phạm vi ngày (từ ngày bắt đầu đến ngày kết thúc). Đây là một công cụ linh hoạt và tiện dụng dành cho cả Android và iOS, giúp việc chọn ngày trong ứng dụng trở nên đơn giản hơn.

### Cách triển khai Calendar Picker

Trong ví dụ này, chúng ta sẽ tạo một Calendar Picker có tính năng chọn phạm vi ngày. Khi người dùng chọn ngày bắt đầu và ngày kết thúc, các ngày này sẽ được hiển thị dưới dạng văn bản. Để thực hiện, chúng ta sẽ bắt đầu với việc tạo một dự án React Native mới.

#### Khởi tạo ứng dụng React Native

Để bắt đầu, hãy sử dụng npx react-native để khởi tạo dự án. Nếu đã cài đặt gói react-native-cli toàn cục trước đó, hãy gỡ cài đặt để tránh các lỗi không mong muốn bằng lệnh npm uninstall -g react-native-cli @react-native-community/cli. Sau đó, dùng lệnh npx react-native init ProjectName để khởi tạo dự án. Nếu muốn dùng một phiên bản React Native cụ thể, có thể thêm --version vào lệnh khởi tạo, ví dụ: npx react-native init ProjectName --version X.XX.X.

#### Cài đặt các thư viện phụ thuộc

Để sử dụng Calendar Picker, cần cài đặt thư viện react-native-calendar-picker. Đầu tiên, di chuyển vào thư mục dự án bằng lệnh cd ProjectName, rồi cài đặt các thư viện cần thiết:

* npm install react-native-calendar-picker --save
* npm install moment –save

#### Code mẫu cho Calendar Picker

Dưới đây là một ví dụ sử dụng Calendar Picker để hiển thị ngày bắt đầu và ngày kết thúc. Trong tệp App.js, import các thành phần cần thiết và cấu hình Calendar Picker như sau:

Bảng : Mã nguồn cách sử dụng react-native-calendar-picker

|  |
| --- |
| import React, { useState } from 'react';  import { SafeAreaView, StyleSheet, View, Text } from 'react-native';  import CalendarPicker from 'react-native-calendar-picker';  const App = () => {  const [selectedStartDate, setSelectedStartDate] = useState(null);  const [selectedEndDate, setSelectedEndDate] = useState(null);  const onDateChange = (date, type) => {  if (type === 'END\_DATE') {  setSelectedEndDate(date);  } else {  setSelectedEndDate(null);  setSelectedStartDate(date);  }  };  return (  <SafeAreaView style={styles.container}>  <View style={styles.container}>  <Text style={styles.titleStyle}>React Native Calendar Picker</Text>  <CalendarPicker  startFromMonday={true}  allowRangeSelection={true}  minDate={new Date(2018, 1, 1)}  maxDate={new Date(2050, 6, 3)}  weekdays={['Mon', 'Tue', 'Wed', 'Thu', 'Fri', 'Sat', 'Sun']}  months={['January', 'February', 'March', 'April', 'May', 'June', 'July', 'August', 'September', 'October', 'November', 'December']}  previousTitle="Previous"  nextTitle="Next"  todayBackgroundColor="#e6ffe6"  selectedDayColor="#66ff33"  selectedDayTextColor="#000000"  scaleFactor={375}  textStyle={{ fontFamily: 'Cochin', color: '#000000' }}  onDateChange={onDateChange}  />  <View style={styles.textStyle}>  <Text>Selected Start Date: {selectedStartDate ? selectedStartDate.toString() : ''}</Text>  <Text>Selected End Date: {selectedEndDate ? selectedEndDate.toString() : ''}</Text>  </View>  </View>  </SafeAreaView>  );  };  export default App;  const styles = StyleSheet.create({  container: { flex: 1, paddingTop: 30, backgroundColor: '#ffffff', padding: 16 },  textStyle: { marginTop: 10 },  titleStyle: { textAlign: 'center', fontSize: 20, margin: 20 },  }); |

#### Chạy ứng dụng React Native

Sau khi cấu hình xong, mở terminal và điều hướng tới thư mục dự án cd ProjectName. Để chạy trên thiết bị ảo Android hoặc iOS, mở terminal mới và sử dụng lệnh npx react-native run-android cho Android, hoặc npx react-native run-ios cho iOS (chỉ áp dụng trên macOS).

*Kết quả*

A screenshot of a calendar

Description automatically generated

Hình : Kết quả sau khi cài đặt và sử dụng thư viện react-native-calendar-picker

## Calendar with Events [8]

* + 1. **Giới thiệu**

Thành phần EventCalendar giúp hiển thị các sự kiện dưới dạng một mảng JSON đơn giản, bao gồm ngày giờ và nội dung sự kiện. Khi bấm vào một sự kiện trong lịch, có thể xem thêm chi tiết. Nếu muốn lưu sự kiện vào lịch của thiết bị, có thể tham khảo các gói hỗ trợ tính năng này cho cả Android và iOS.

* + 1. **Cách triển khai**

Dưới đây là cách cấu hình cơ bản của EventCalendar để hiển thị sự kiện:

Bảng : Mã nguồn cấu hình EventCalendar

|  |
| --- |
| <EventCalendar  eventTapped={this.eventClicked.bind(this)}  events={this.state.events}  width={width}  size={60}  initDate={'2019-01-01'}  scrollToFirst  /> |

Có thể tùy chỉnh hiển thị sự kiện, số lượng ngày hiển thị, và ngày bắt đầu với các thuộc tính của EventCalendar.

#### Cài đặt thư viện

Đầu tiên, tạo một dự án mới trong React Native nếu chưa có:

* npx react-native init ProjectName

Để cài đặt thư viện react-native-events-calendar, sử dụng các lệnh sau:

* cd ProjectName
* npm install react-native-events-calendar --save

#### Ví dụ Code cho Calendar với Sự kiện

Trong tệp App.js, sử dụng đoạn mã sau để tạo một Calendar với danh sách sự kiện mẫu:

Bảng : Mã nguồn cách sử dụng Calendar với Sự kiện

|  |
| --- |
| import React, { useState } from 'react';  import { SafeAreaView, StyleSheet, View, Dimensions } from 'react-native';  import EventCalendar from 'react-native-events-calendar';  let { width } = Dimensions.get('window');  const App = () => {  const [events, setEvents] = useState([  {  start: '2020-01-01 00:00:00',  end: '2020-01-01 02:00:00',  title: 'New Year Party',  summary: 'xyz Location',  },  {  start: '2020-01-01 01:00:00',  end: '2020-01-01 02:00:00',  title: 'New Year Wishes',  summary: 'Call to every one',  },  {  start: '2020-01-02 00:30:00',  end: '2020-01-02 01:30:00',  title: 'Parag Birthday Party',  summary: 'Call him',  },  {  start: '2020-01-03 01:30:00',  end: '2020-01-03 02:20:00',  title: 'My Birthday Party',  summary: 'Lets Enjoy',  },  {  start: '2020-02-04 04:10:00',  end: '2020-02-04 04:40:00',  title: 'Engg Expo 2020',  summary: 'Expoo Vanue not confirm',  },  ]);  const eventClicked = (event) => {  alert(JSON.stringify(event));  };  return (  <SafeAreaView style={styles.container}>  <View style={styles.container}>  <EventCalendar  eventTapped={eventClicked}  events={events}  width={width}  size={60}  initDate={'2020-01-01'}  scrollToFirst  />  </View>  </SafeAreaView>  );  };  export default App;  const styles = StyleSheet.create({  container: {  flex: 1,  alignItems: 'center',  justifyContent: 'center',  },  }); |

#### Chạy ứng dụng

* Khởi động Metro Bundler:
* npx react-native start
* Chạy ứng dụng trên thiết bị Android hoặc iOS:
* Đối với Android: npx react-native run-android
* Đối với iOS (chỉ trên macOS): npx react-native run-ios

*Kết quả*

*A screenshot of a calendar

Description automatically generatedA screenshot of a message

Description automatically generated*A screenshot of a message

Description automatically generated

Hình : Kết quả chạy ứng dụng sử dụng react-native-events-calendar

## Add Event in Device’s Calendar [9]

### Giới thiệu

Đây là hướng dẫn về cách thêm sự kiện vào lịch của thiết bị từ ứng dụng React Native. Thư viện react-native-add-calendar-event sẽ mở một activity trên Android hoặc hiển thị một cửa sổ modal trên iOS để thêm, xem, hoặc chỉnh sửa sự kiện trong lịch của thiết bị, giúp quá trình này trở nên đơn giản hơn.

Bảng : Cấu hình Add Event in Device’s Calendar

|  |
| --- |
| const eventConfig = {  title,  startDate: utcDateToString(startDateUTC),  endDate: utcDateToString(moment.utc(startDateUTC).add(1, 'hours')),  notes: 'tasty!',  navigationBarIOS: {  tintColor: 'orange',  backgroundColor: 'green',  titleColor: 'blue',  },  };  AddCalendarEvent.presentEventCreatingDialog(eventConfig)  .then((eventInfo) => {  alert('eventInfo -> ' + JSON.stringify(eventInfo));  })  .catch((error) => {  alert('Error -> ' + error);  }); |

Ví dụ này sẽ hiển thị tiêu đề và thời gian của sự kiện cần thêm lên đầu màn hình. Thời gian sự kiện sẽ là thời điểm hiện tại để dễ kiểm tra thông báo sự kiện. Một nút “Add Event to Calendar” sẽ thực hiện việc thêm sự kiện vào lịch khi nhấn.

Sau khi thêm sự kiện, ứng dụng sẽ hiển thị chi tiết sự kiện với ID. Người dùng cũng có thể nhập ID này để chỉnh sửa hoặc xem sự kiện.

### Cách triển khai

Khởi tạo một dự án React Native mới với lệnh sau:

* npx react-native init ProjectName

#### Cài đặt thư viện

* Cài đặt react-native-add-calendar-event để thêm sự kiện vào lịch:
* npm install react-native-add-calendar-event --save
* Cài đặt moment.js để quản lý thời gian:
* npm install moment –save

Trên iOS, cài đặt CocoaPods bằng lệnh:

* npx pod-install

#### Định dạng thời gian cho sự kiện

Thời gian bắt đầu và kết thúc của sự kiện nên ở định dạng UTC chuẩn:

* **startDate** - UTC, định dạng: 'YYYY-MM-DDTHH:mm:ss.SSSZ'
* **endDate** - UTC, định dạng: 'YYYY-MM-DDTHH:mm:ss.SSSZ'

Lấy thời gian hiện tại theo định dạng UTC với moment.js:

* const TIME\_NOW\_IN\_UTC = moment.utc();

#### Code để thêm sự kiện vào lịch thiết bị

Thay thế nội dung trong App.js với mã dưới đây:

Bảng : Mã nguồn để thêm sự kiện vào lịch thiết bị

|  |
| --- |
| import React, {useState} from 'react';  // import all the components we are going to use  import {  SafeAreaView,  StyleSheet,  Text,  View,  TextInput,  TouchableOpacity,  } from 'react-native';  //Import library for AddCalendarEvent  import \* as AddCalendarEvent from 'react-native-add-calendar-event';  //Import moment.js to deal with time  import moment from 'moment';  const EVENT\_TITLE = 'Lunch';  const TIME\_NOW\_IN\_UTC = moment.utc();  const utcDateToString = (momentInUTC) => {  let s = moment.utc(momentInUTC)  .format('YYYY-MM-DDTHH:mm:ss.SSS[Z]');  return s;  };  const addToCalendar = (title, startDateUTC) => {  const eventConfig = {  title,  startDate: utcDateToString(startDateUTC),  endDate:  utcDateToString(moment.utc(startDateUTC).add(1, 'hours')),  notes: 'tasty!',  navigationBarIOS: {  tintColor: 'orange',  backgroundColor: 'green',  titleColor: 'blue',  },  };  AddCalendarEvent.presentEventCreatingDialog(eventConfig)  .then((eventInfo) => {  alert('eventInfo -> ' + JSON.stringify(eventInfo));  })  .catch((error) => {  // handle error such as when user rejected permissions  alert('Error -> ' + error);  });  };  const editCalendarEventWithId = (eventId) => {  if (!eventId) {  alert('Please Insert Event Id');  return;  }  const eventConfig = {  eventId,  };  AddCalendarEvent.presentEventEditingDialog(eventConfig)  .then((eventInfo) => {  alert('eventInfo -> ' + JSON.stringify(eventInfo));  })  .catch((error) => {  alert('Error -> ' + error);  });  };  const showCalendarEventWithId = (eventId) => {  if (!eventId) {  alert('Please Insert Event Id');  return;  }  const eventConfig = {  eventId,  allowsEditing: true,  allowsCalendarPreview: true,  navigationBarIOS: {  tintColor: 'orange',  backgroundColor: 'green',  },  };  AddCalendarEvent.presentEventViewingDialog(eventConfig)  .then((eventInfo) => {  alert('eventInfo -> ' + JSON.stringify(eventInfo));  })  .catch((error) => {  alert('Error -> ' + error);  });  };  const App = () => {  const [text, setText] = useState('');  return (  <SafeAreaView style={styles.container}>  <View style={styles.container}>  <Text style={styles.titleStyle}>  Example to Add Event in Google Calendar  from React Native App  </Text>  <Text style={styles.heading}>  Event title: {EVENT\_TITLE}  {'\n'}  Event Date Time: {  moment.utc(TIME\_NOW\_IN\_UTC).local().format('lll')  }  </Text>  <TouchableOpacity  style={[styles.buttonStyle, {minWidth: '100%'}]}  onPress={() => {  addToCalendar(EVENT\_TITLE, TIME\_NOW\_IN\_UTC);  }}>  <Text style={styles.buttonTextStyle}>  Add Event to Calendar  </Text>  </TouchableOpacity>  <TextInput  style={styles.inputStyle}  placeholder="enter event id"  onChangeText={(text) => setText(text)}  value={text}  />  <View style={{flexDirection: 'row'}}>  <TouchableOpacity  style={styles.buttonStyle}  onPress={() => {  editCalendarEventWithId(text);  }}>  <Text style={styles.buttonTextStyle}>  Edit Event  </Text>  </TouchableOpacity>  <View style={{margin: 5}} />  <TouchableOpacity  style={styles.buttonStyle}  onPress={() => {  showCalendarEventWithId(text);  }}>  <Text style={styles.buttonTextStyle}>  View Event  </Text>  </TouchableOpacity>  </View>  </View>  </SafeAreaView>  );  };  export default App;  const styles = StyleSheet.create({  container: {  flex: 1,  alignItems: 'center',  justifyContent: 'center',  backgroundColor: '#307ecc',  padding: 16,  },  heading: {  color: 'white',  fontSize: 16,  textAlign: 'center',  margin: 10,  },  buttonStyle: {  paddingVertical: 10,  paddingHorizontal: 30,  backgroundColor: '#f5821f',  margin: 15,  },  buttonTextStyle: {  color: 'white',  textAlign: 'center',  },  buttonHalfStyle: {  alignItems: 'center',  backgroundColor: '#DDDDDD',  padding: 10,  flex: 1,  },  titleStyle: {  color: 'white',  textAlign: 'center',  fontSize: 20,  marginBottom: 20,  },  inputStyle: {  height: 40,  minWidth: '100%',  marginBottom: 10,  marginTop: 30,  padding: 10,  backgroundColor: '#ffe6e6',  },  }); |

#### Chạy ứng dụng

* Mở terminal, di chuyển vào thư mục dự án và chạy Metro Bundler:
* npx react-native start
* Mở một terminal mới và chạy ứng dụng:
* Đối với Android: npx react-native run-android
* Đối với iOS (chỉ trên macOS): npx react-native run-ios
* Đối với expo: npx expo start

*Kết quả*

A screenshot of a calendar

Description automatically generatedA screenshot of a calendar

Description automatically generatedA screenshot of a phone

Description automatically generated

A screenshot of a phone

Description automatically generatedA screenshot of a calendar

Description automatically generatedA screenshot of a phone

Description automatically generated

Hình : Kết quả chạy ứng dụng với các sự kiện Add Event in Device’s Calendar

# Tài liệu tham khảo

[1] S. Agrawal, “Example of Searchable Dropdown / Picker in React Native”, About React. Truy cập: 1 Tháng Mười-Một 2024. [Online]. Available at: https://aboutreact.com/example-of-searchable-dropdown-picker-in-react-native/

[2] S. Agrawal, “Multiple Select / Dropdown / Picker Example in React Native”, About React. Truy cập: 1 Tháng Mười-Một 2024. [Online]. Available at: https://aboutreact.com/multiple-select-dropdown-picker-example-in-react-native/

[3] S. Agrawal, “React Native Date Picker - To Pick the Date using Native Calendar”, About React. Truy cập: 31 Tháng Mười 2024. [Online]. Available at: https://aboutreact.com/react-native-datepicker/

[4] S. Agrawal, “React Native Time Picker – Pick the Time using Native Time Picker”, About React. Truy cập: 31 Tháng Mười 2024. [Online]. Available at: https://aboutreact.com/react-native-timepicker/

[5] S. Agrawal, “React Native CountDown Timer|react-native-countdown-component”, About React. Truy cập: 31 Tháng Mười 2024. [Online]. Available at: https://aboutreact.com/react-native-countdown-timer/

[6] S. Agrawal, “Example Code of React Native Timer and Stopwatch - About React Native”, About React. Truy cập: 31 Tháng Mười 2024. [Online]. Available at: https://aboutreact.com/react-native-timer-stopwatch/

[7] S. Agrawal, “React Native Calendar Picker using react-native-calendar-picker”, About React. Truy cập: 31 Tháng Mười 2024. [Online]. Available at: https://aboutreact.com/react-native-calendar-picker/

[8] S. Agrawal, “Example of Calendar with Events Listed in React Native”, About React. Truy cập: 31 Tháng Mười 2024. [Online]. Available at: https://aboutreact.com/example-of-calendar-with-events-listed-in-react-native/

[9] S. Agrawal, “Add Event in Device’s Calendar from React Native App”, About React. Truy cập: 31 Tháng Mười 2024. [Online]. Available at: https://aboutreact.com/react-native-add-event-in-device-calendar/