**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 SEMINAR**

**MÔN CÔNG NGHỆ LẬP TRÌNH ĐA NỀN TẢNG CHO ỨNG DỤNG DI DỘNG**

**Chuyên đề:**

**TÌM HIỂU VỀ CHARTS AND GRAPH, HYPERLINKING VÀ ENCODING**

**GVHD: ThS. Võ Ngọc Tân**

**Nhóm sinh viên thực hiện**:

1. **Đỗ Hoàng Yến MSSV: 23540050**
2. **Lê Ngoạn MSSV: 22540013**

**🙢 Tp. Hồ Chí Minh, 11/2024 🙠**

# NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

*……., ngày……...tháng……năm 2024*

**Người nhận xét**

*(Ký tên và ghi rõ họ tên****)***

# BẢNG PHÂN CÔNG, ĐÁNH GIÁ THÀNH VIÊN

|  |  |  |  |
| --- | --- | --- | --- |
| **Họ và tên** | **MSSV** | **Phân công** | **Đánh giá** |
| Đỗ Hoàng Yến | 23540050 |  | Hoàn thành tốt |
| Lê Ngoạn | 22540013 |  | Hoàn thành tốt |

**Mục Lục**

[NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN 2](#_Toc181719039)

[BẢNG PHÂN CÔNG, ĐÁNH GIÁ THÀNH VIÊN 3](#_Toc181719040)

[LỜI MỞ ĐẦU 6](#_Toc181719041)

[Chương 1. CHARTS AND GRAPH 7](#_Toc181719042)

[1.1. Speedometer 7](#_Toc181719043)

[1.2. RN Pie Chart 9](#_Toc181719044)

[1.3. React Native Chart Kit 12](#_Toc181719045)

[Chương 2. HYPERLINKING 15](#_Toc181719046)

[2.1. Display Hyperlink 15](#_Toc181719047)

[2.2. RN Linking 18](#_Toc181719048)

[Chương 3. ENCODING 20](#_Toc181719049)

[3.1. Generate QR Code 20](#_Toc181719050)

[3.2. MD5 Encoding 22](#_Toc181719051)

[3.3. SHA256 Encoding 25](#_Toc181719052)

[Chương 4. PROFANITY FILTER 28](#_Toc181719053)

[Chương 5. KẾT LUẬN 31](#_Toc181719054)

[TÀI LIỆU THAM KHẢO 32](#_Toc181719055)

**Mục lục hình ảnh**

[Hình 1: Mẫu chạy thử biểu đồ tốc kế trên React Native. 8](#_Toc181719002)

[Hình 2: Mẫu chạy thử biểu đồ tròn trên React Native 11](#_Toc181719003)

[Hình 3: Mẫu chạy thử sử dụng thư viện React Native Chart Kit. 14](#_Toc181719004)

[Hình 4: Mẫu thử Hyperlink trên React Native 1. 16](#_Toc181719005)

[Hình 6: Mẫu thử Hyperlink trên React Native 2. 17](#_Toc181719006)

[Hình 7: Mẫu thử RN Linking trên React Native. 19](#_Toc181719007)

[Hình 8: Mẫu thử QR Code trên React Native. 21](#_Toc181719008)

[Hình 9: Mẫu chạy thử MD5 Encoding trên React Native 1. 23](#_Toc181719009)

[Hình 10: Mẫu chạy thử MD5 Encoding trên React Native 2. 24](#_Toc181719010)

[Hình 11: Mẫu chạy thử SHA256 Encoding trên React Native 1. 26](#_Toc181719011)

[Hình 12: Mẫu chạy thử SHA256 Encoding trên React Native 2. 27](#_Toc181719012)

[Hình 13: Mẫu chạy thử IV. Profanity Filter trên React Native. 30](#_Toc181719013)

# LỜI MỞ ĐẦU

React Native ra đời vào năm 2015 cho phép lập trình viên xây dựng ứng dụng di động cho iOS và Android bằng JavaScript và React với hiệu suất gần tương đương với ứng dụng được viết bằng ngôn ngữ gốc, để giải quyết vấn đề của việc phát triển ứng dụng di động truyền thống (yêu cầu viết mã riêng cho từng nền tảng). Bài báo cáo này xin được phép trích chọn ba tính năng quan trọng của React Native để giới thiệu: biểu đồ và đồ thị (Charts and Graph), liên kết siêu văn bản (Hyperlinking) , và mã hóa (Encoding).

# CHARTS AND GRAPH

## Speedometer

**Tổng quan**: Biểu đồ Speedometer (hoặc biểu đồ tốc kế) là một dạng biểu đồ đo lường thường dùng để biểu diễn giá trị trên một thang điểm từ thấp đến cao. Đồ thị này mô phỏng thiết kế của đồng hồ tốc độ trong xe hơi, với kim chỉ và vùng màu sắc tương ứng với các giá trị khác nhau. Speedometer Graph rất hiệu quả trong việc trực quan hóa dữ liệu khi muốn hiển thị trạng thái hiện tại hoặc hiệu suất trên một thang đo cụ thể.

**Đặc điểm** của biểu đồ Speedometer

* Dễ đọc: Speedometer có cách hiển thị rõ ràng với các vùng màu và kim chỉ báo, giúp người dùng nhanh chóng nhận diện được mức độ hoặc trạng thái dữ liệu.
* Trực quan: Các vùng màu (xanh, vàng, đỏ) thường được dùng để phân biệt các mức độ từ thấp đến cao, hoặc trạng thái từ an toàn đến cảnh báo, nguy hiểm.
* Phổ biến trong theo dõi hiệu suất: Được sử dụng trong các ứng dụng quản lý hiệu suất, ứng dụng IoT, và các dashboard tài chính để biểu thị chỉ số quan trọng một cách dễ hiểu.

Speedometer Graph được sử dụng trong nhiều lĩnh vực:

* Đo lường hiệu suất: Theo dõi hiệu suất công việc, hiệu suất của máy móc, hoặc các chỉ số KPI.
* Phân tích tài chính: Giúp hiển thị mức độ đầu tư, độ rủi ro, hoặc trạng thái tài chính hiện tại.
* Ứng dụng IoT: Giám sát các thông số như nhiệt độ, độ ẩm, hoặc mức tiêu thụ điện năng.
* Ứng dụng sức khỏe: Theo dõi nhịp tim, chỉ số BMI, hoặc bất kỳ chỉ số sức khỏe nào cần hiển thị trên một thang đo cụ thể.

**Cách sử dụng**

Cài thư viện:

*npm install react-native-speedometer*

*yarn add react-native-speedometer*

Nhập Component Speedometer từ thư viện react-native-speedometer

import Speedometer from "react-native-speedometer";

Cài đặt các thuộc tính :

  <Speedometer

        value={70} // Giá trị hiện tại

        maxValue={100} // Giá trị tối đa

        minValue={0} // Giá trị tối thiểu

        internalColor={"green"} // Màu bên trong

        outerColor={"lightgray"} // Màu bên ngoài

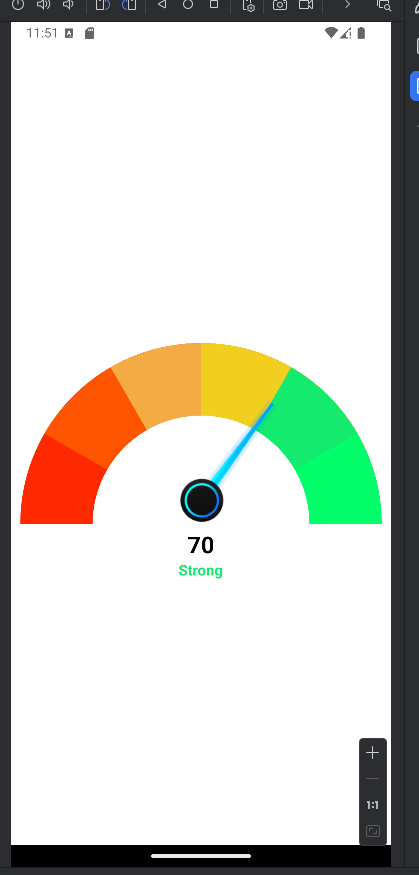
        needleColor={"red"} // Màu kim

        needleTransitionDuration={100} // Thời gian chuyển động của kim

        needleTransitionTimingFunction={"ease-in-out"} // Hàm chuyển động

      />

**Chạy thử**



Hình 1: Mẫu chạy thử biểu đồ tốc kế trên React Native.

## [RN Pie Chart](https://aboutreact.com/react-native-pie-chart)

**Tổng quan:** Thư viện mã nguồn mở với hai biến thể để hiển thị dữ liệu dưới dạng biểu đồ hình tròn. Giữ kích thước gói nhỏ và dễ sử dụng.

**Đặc điểm**

* Trực quan và dễ hiểu: Biểu đồ tròn thể hiện tỷ lệ phần trăm của mỗi phần trong tổng thể một cách rõ ràng, giúp người dùng nhanh chóng nhận diện cấu trúc phân bố dữ liệu.
* Tính tùy biến cao: Các thư viện hỗ trợ biểu đồ tròn thường cho phép người dùng tùy chỉnh nhiều thuộc tính như màu sắc, kích thước, nhãn và hình dạng của các phần trong biểu đồ.
* Hỗ trợ tương tác: Nhiều thư viện cho phép thêm tính năng tương tác như nhấn vào từng phần để hiển thị thông tin chi tiết hoặc để làm nổi bật phần đó.
* Hiệu suất tốt: Biểu đồ tròn được tối ưu hóa cho hiệu suất, cho phép hiển thị mượt mà ngay cả khi có nhiều phần.
* Dễ dàng tích hợp: RN Pie Chart có thể dễ dàng tích hợp vào các ứng dụng React Native, với cấu trúc mã rõ ràng và dễ sử dụng.
* Thư viện hỗ trợ: react-native-chart-kit, victory-native, react-native-svg.

**Cách sử dụng:**

Cài thư viện:

npm install react-native-chart-kit react-native-svg

yarn add react-native-chart-kit react-native-svg

Nhập component PieChart từ thư viện react-native-chart-kit

import { PieChart } from 'react-native-chart-kit';

Cài đặt thuộc tính

const data = [

  {

    name: "Phần A",

    population: 50,

    color: "#ff6384",

    legendFontColor: "#7F7F7F",

    legendFontSize: 15,

  },

  {

    name: "Phần B",

    population: 30,

    color: "#36a2eb",

    legendFontColor: "#7F7F7F",

    legendFontSize: 15,

  },

  {

    name: "Phần C",

    population: 20,

    color: "#cc65fe",

    legendFontColor: "#7F7F7F",

    legendFontSize: 15,

  },

];

const App = () => {

  return (

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

      <PieChart

        data={data}

        width={Dimensions.get("window").width - 40} // Kích thước biểu đồ

        height={220} // Chiều cao biểu đồ

        chartConfig={{

          backgroundColor: "#ffffff",

          color: (opacity = 1) => `rgba(0, 0, 0, ${opacity})`,

          labelColor: (opacity = 1) => `rgba(0, 0, 0, ${opacity})`,

        }}

        accessor="population" // Trường dữ liệu cho các phần

        backgroundColor="transparent" // Màu nền

        paddingLeft="15" // Khoảng cách bên trái

      />

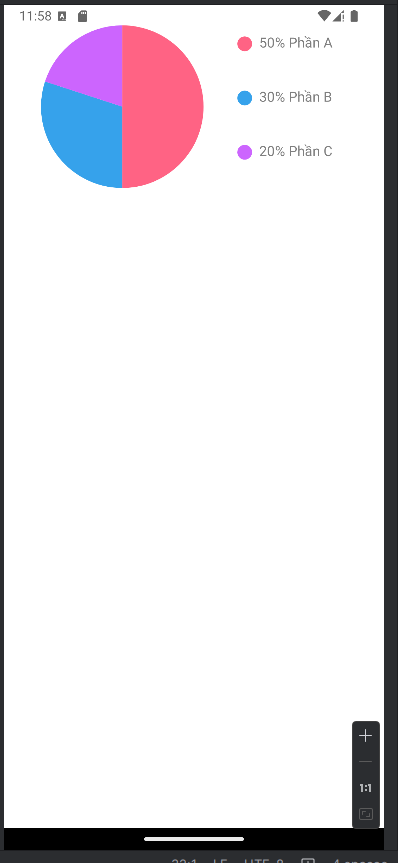
    </View>

  );

};

export default App;

**Chạy thử**



Hình 2: Mẫu chạy thử biểu đồ tròn trên React Native

## [React Native Chart Kit](https://aboutreact.com/react-native-chart-kit)

**Tổng quan:** Là một thư viện mạnh mẽ và dễ sử dụng cho việc tạo biểu đồ và đồ thị trong các ứng dụng React Native. Thư viện này giúp các nhà phát triển dễ dàng trực quan hóa dữ liệu bằng cách cung cấp nhiều loại biểu đồ khác nhau với cấu hình linh hoạt.

**Đặc điểm**

* Dễ sử dụng: Thư viện có API thân thiện, dễ hiểu và tài liệu hướng dẫn chi tiết, giúp việc triển khai trở nên nhanh chóng.
* Tùy biến linh hoạt: Các thuộc tính biểu đồ có thể được tùy chỉnh để phù hợp với yêu cầu của dự án, bao gồm màu sắc, nhãn, kích thước, và nhiều hơn nữa.
* Tích hợp tốt: React Native Chart Kit có thể dễ dàng tích hợp với các ứng dụng hiện có mà không gặp phải vấn đề tương thích.
* Hiệu suất tốt: Thư viện được tối ưu hóa để hoạt động mượt mà trên cả thiết bị iOS và Android.

**Cách sử dụng**

Cài đặt thư viện:

npm install react-native-chart-kit react-native-svg

yarn add react-native-chart-kit react-native-svg

Import component LineChart từ thư viện react-native-chart-kit

import { LineChart } from 'react-native-chart-kit';

Cài đặt thuộc tính

  <LineChart

        data={data}

        width={Dimensions.get('window').width - 30} // width of the chart

        height={220} // height of the chart

        chartConfig={{

          backgroundColor: "#ffffff",

          backgroundGradientFrom: "#ffffff",

          backgroundGradientTo: "#ffffff",

          decimalPlaces: 2, // optional, defaults to 2dp

          color: (opacity = 1) => `rgba(0, 0, 255, ${opacity})`,

          labelColor: (opacity = 1) => `rgba(0, 0, 0, ${opacity})`,

          style: {

            borderRadius: 16,

          },

        }}

        style={{

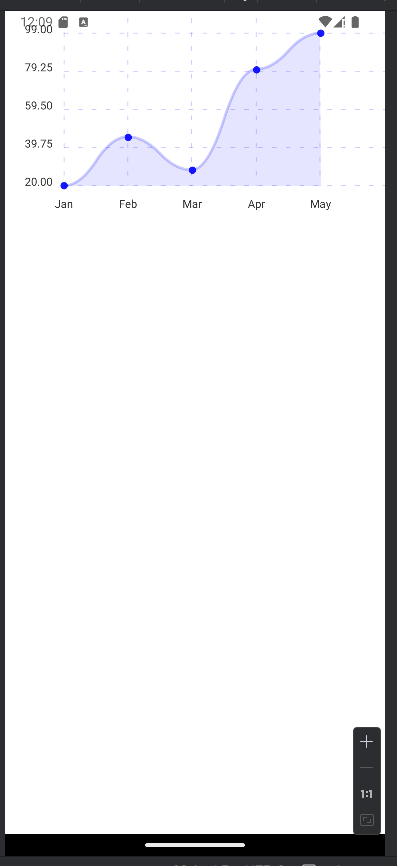
          marginVertical: 8,

          borderRadius: 16,

        }}

      />

**Chạy thử**



Hình 3: Mẫu chạy thử sử dụng thư viện React Native Chart Kit.

# HYPERLINKING

## [Display Hyperlink](https://aboutreact.com/react-native-hyperlink)

**Tổng quan:** Display Hyperlink là cách mà một liên kết được trình bày và hiển thị trên giao diện người dùng, bao gồm các thuộc tính như màu sắc, kích thước, hình thức và trạng thái khi người dùng tương tác với nó.

**Chức năng**

* Chuyển hướng: Liên kết cho phép người dùng di chuyển từ một trang web hoặc màn hình này sang một trang khác, cung cấp khả năng truy cập nhanh đến nội dung hoặc chức năng khác.
* Tương tác: Liên kết có thể dẫn đến các hành động như mở một trang mới, tải xuống tài liệu, hoặc mở một modal trong ứng dụng.

**Các thuộc tính quan trọng**

* Màu sắc: Thường sử dụng màu sắc khác biệt để người dùng dễ nhận biết rằng đó là một liên kết.
* Gạch chân: Liên kết thường được gạch chân để phân biệt với văn bản thông thường.
* Trạng thái hover: Khi di chuột hoặc nhấn vào liên kết, có thể có sự thay đổi về màu sắc hoặc kiểu dáng để người dùng biết rằng họ đang tương tác với một liên kết.

**Cách cài đặt**

Import Linking từ thư viện react-native

Cài đặt URL cho Linking

*Linking.openURL('https://www.google.com');*

*const App = () => {*

*const handlePress = () => {*

*Linking.openURL("https://www.google.com");*

*};*

*return (*

*<View style={styles.container}>*

*<Text style={styles.link} onPress={handlePress}>*

*Visit Example.com*

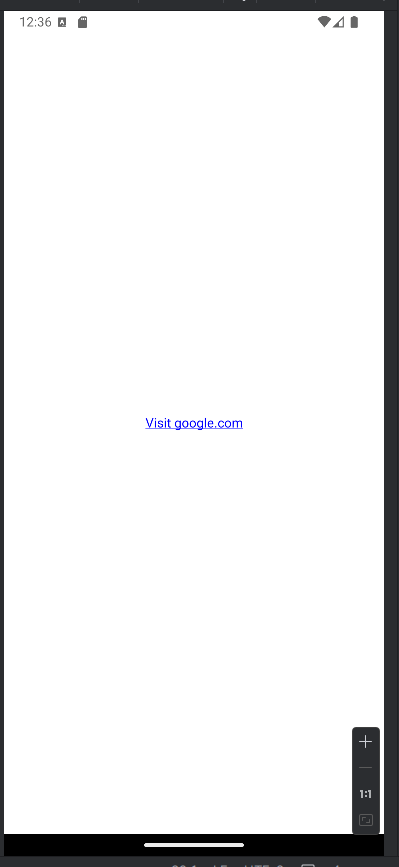
*</Text>*

*</View>*

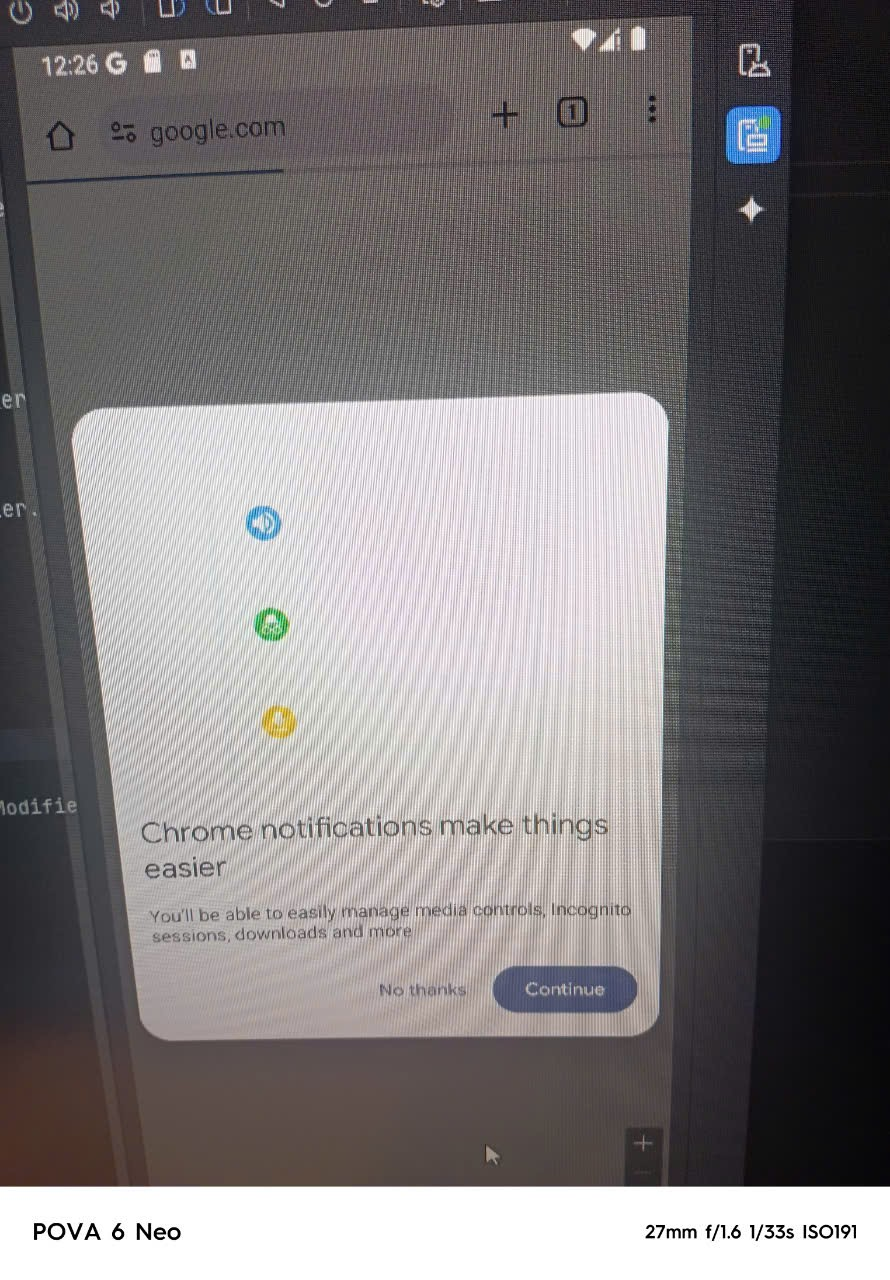
*);*

*};*

**Chạy thử**



Hình 4: Mẫu thử Hyperlink trên React Native 1.



Hình 6: Mẫu thử Hyperlink trên React Native 2.

## [RN Linking](https://aboutreact.com/react-native-clicking-on-a-text-link-to-open-it-in-the-browser-using-linking)

**Tổng quan:** RN Linking là một module được sử dụng để xử lý các liên kết URL, cả từ các liên kết bên ngoài mở ứng dụng (deep linking) và từ trong ứng dụng để điều hướng hoặc mở liên kết bên ngoài (như website hoặc ứng dụng khác).

**Đặc điểm:**

Gồm các khả năng và tính năng nổi bật giúp quản lý và xử lý các liên kết hiệu quả trong ứng dụng di động.

Hỗ trợ đa nền tảng

Cross-Platform: RN Linking hoạt động trên cả iOS và Android, giúp đồng nhất trong cách xử lý liên kết giữa các nền tảng mà không cần viết mã riêng cho từng hệ điều hành.

**Cách cài đặt**

Import Linking từ thư viện react-native

import { View, Button, Linking, Alert } from "react-native";

Cài đặt cho chức năng gửi SMS

const App = () => {

const sendSMS = async () => {

const phoneNumber = "0962708976"; // Số điện thoại bạn muốn gửi

const message = "Chào bạn! Đây là một tin nhắn từ ứng dụng React Native.";

// Định dạng URL cho tin nhắn

const url = `sms:${phoneNumber}?body=${encodeURIComponent(message)}`;

const supported = await Linking.canOpenURL(url);

if (supported) {

await Linking.openURL(url);

} else {

Alert.alert("Lỗi", "Không thể mở ứng dụng gửi tin nhắn.");

}

};

return (

<View style={{ padding: 20 }}>

<Button title="Gửi Tin Nhắn" onPress={sendSMS} />

</View>

);

};

**Chạy thử**

A screen shot of a phone

Description automatically generated

Hình 7: Mẫu thử RN Linking trên React Native.

# ENCODING

## [Generate QR Code](https://aboutreact.com/generation-of-qr-code-in-react-native)

**Tổng quan:** QR Code (Quick Response Code) là một dạng mã vạch hai chiều thường được sử dụng để lưu trữ và chia sẻ thông tin một cách nhanh chóng. Trong ứng dụng React Native, việc tạo QR Code có thể giúp người dùng dễ dàng quét và chia sẻ thông tin, chẳng hạn như URL, thông tin liên lạc, hoặc bất kỳ dữ liệu nào khác.

**Đặc điểm:**

* Chia sẻ Thông tin Nhanh chóng: QR Code cho phép người dùng quét và truy cập thông tin ngay lập tức.
* Tích hợp Thương mại Điện tử: Có thể sử dụng để tạo mã QR cho sản phẩm, khuyến mãi, hoặc liên kết đến trang thanh toán.

**Cách cài đặt**

Cài đặt thư viện

*npm install react-native-qrcode-svg*

Import QRCode từ thư viện react-native-qrcode-svg

import QRCode from "react-native-qrcode-svg";

Thiết lập thuôc tính cho QRCode

     <QRCode

        value={url}

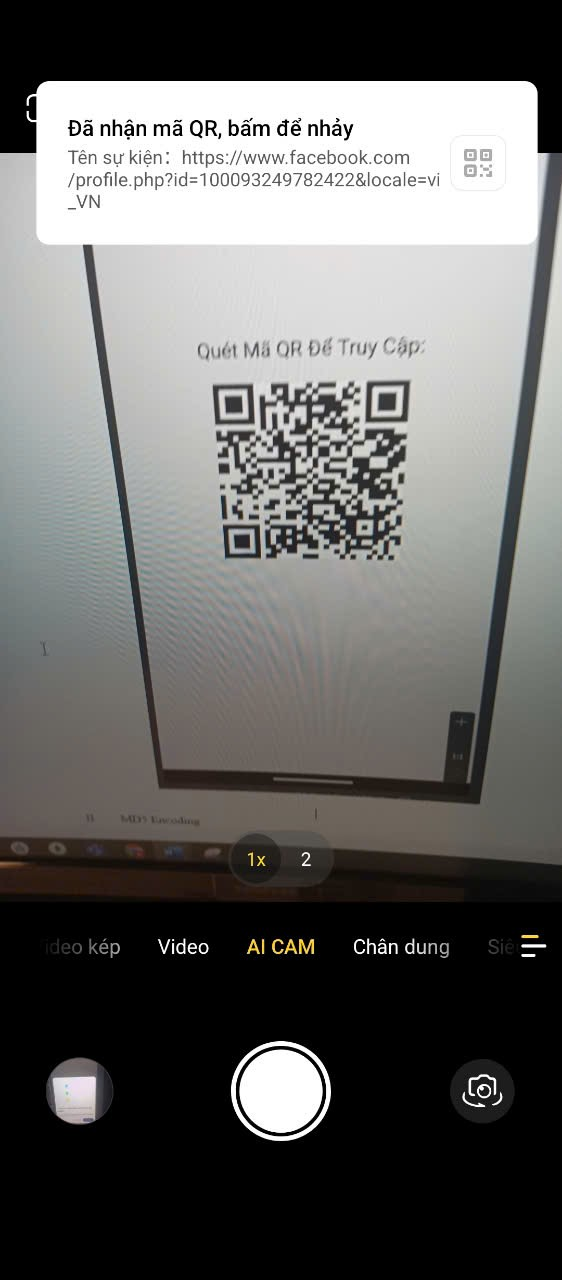
        size={200} // Kích thước mã QR

        color="black" // Màu sắc của mã QR

        backgroundColor="white" // Màu nền của mã QR

      />

**Kết quả:**



Hình 8: Mẫu thử QR Code trên React Native.

## [MD5 Encoding](https://aboutreact.com/md5-in-react-native)

**Tổng quan:** MD5 (Message-Digest Algorithm 5) là một thuật toán băm được sử dụng để tạo ra một chuỗi mã hóa từ dữ liệu đầu vào. Mặc dù MD5 không còn được coi là an toàn cho các ứng dụng cần bảo mật cao, nó vẫn thường được sử dụng cho các mục đích như kiểm tra tính toàn vẹn dữ liệu hoặc tạo mã băm cho chuỗi văn bản.

**Đặc điểm:**

* Kiểm tra Tính Toàn vẹn: MD5 thường được sử dụng để xác minh rằng dữ liệu không bị thay đổi trong quá trình truyền tải.
* Tạo Mã Băm: MD5 có thể được sử dụng để tạo mã băm cho thông tin nhạy cảm như mật khẩu, mặc dù không được khuyến nghị cho các ứng dụng yêu cầu bảo mật cao.

**Cách cài đặt**

Cài đặt thư viện:

*npm install crypto-js*

Cài đặt chương trình

 const handleRegister = () => {

    // Băm mật khẩu và lưu giá trị băm

    const hash = MD5(password).toString();

    setStoredHash(hash);

    Alert.alert("Đăng ký thành công", "Mật khẩu đã được băm và lưu trữ.");

  };

  const handleLogin = () => {

    // Băm mật khẩu nhập vào và so sánh với giá trị đã lưu

    const hash = MD5(password).toString();

    if (hash === storedHash) {

      Alert.alert("Đăng nhập thành công", "Mật khẩu đúng.");

    } else {

      Alert.alert("Đăng nhập thất bại", "Mật khẩu sai.");

    }

  };

  return (

    <View style={styles.container}>

      <TextInput

        style={styles.input}

        placeholder="Nhập mật khẩu"

        secureTextEntry

        value={password}

        onChangeText={setPassword}

      />

      <Button title="Đăng ký" onPress={handleRegister} />

      <Button title="Đăng nhập" onPress={handleLogin} />

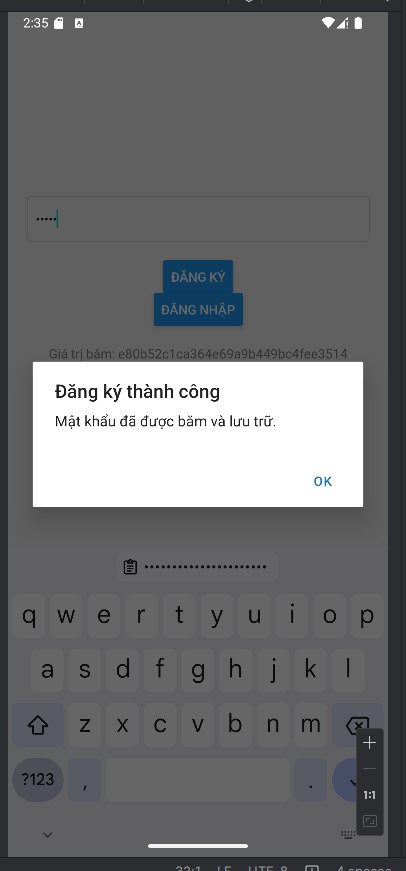
      {storedHash ? (

        <Text style={styles.hash}>Giá trị băm: {storedHash}</Text>

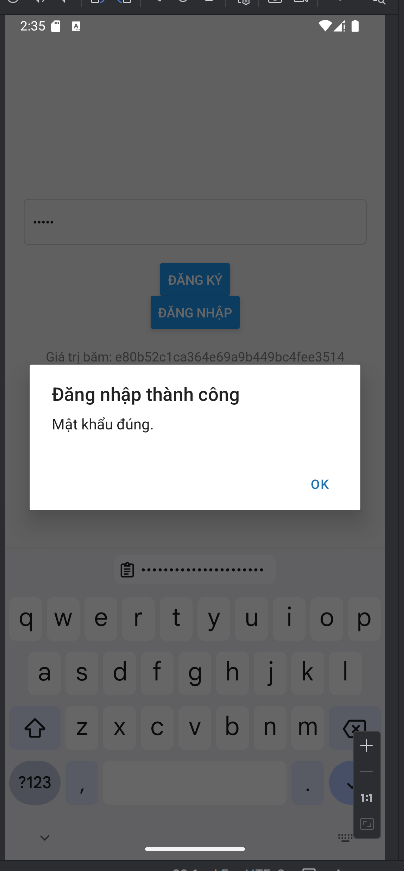
      ) : null}

    </View>

**Chạy thử**



Hình 9: Mẫu chạy thử MD5 Encoding trên React Native 1.



Hình 10: Mẫu chạy thử MD5 Encoding trên React Native 2.

## [SHA256 Encoding](https://aboutreact.com/generate-sha256-encoded-hash-in-react-native)

Tổng quan: SHA-256 (Secure Hash Algorithm 256-bit) là một thuật toán băm an toàn được phát triển bởi Cơ quan An ninh Quốc gia Mỹ (NSA). Nó tạo ra một giá trị băm 256 bit (32 byte) từ dữ liệu đầu vào bất kỳ, thường được sử dụng trong các ứng dụng cần tính bảo mật cao, chẳng hạn như lưu trữ mật khẩu, kiểm tra tính toàn vẹn của dữ liệu, và trong các giao thức bảo mật.

**Đặc điểm**

* Bảo mật Dữ liệu: SHA-256 được sử dụng rộng rãi để bảo vệ mật khẩu và thông tin nhạy cảm bằng cách băm dữ liệu trước khi lưu trữ.
* Kiểm tra Tính Toàn Vẹn: Đảm bảo rằng dữ liệu không bị thay đổi trong quá trình truyền tải hoặc lưu trữ.

**Cách cài đặt**

Cài đặt thư viện:

*npm install crypto-js*

Cài đặt chương trình:

  const [password, setPassword] = useState("");

  const [storedHash, setStoredHash] = useState("");

  const handleRegister = () => {

    // Băm mật khẩu và lưu giá trị băm

    const hash = MD5(password).toString();

    setStoredHash(hash);

    Alert.alert("Đăng ký thành công", "Mật khẩu đã được băm và lưu trữ.");

  };

  const handleLogin = () => {

    // Băm mật khẩu nhập vào và so sánh với giá trị đã lưu

    const hash = MD5(password).toString();

    if (hash === storedHash) {

      Alert.alert("Đăng nhập thành công", "Mật khẩu đúng.");

    } else {

      Alert.alert("Đăng nhập thất bại", "Mật khẩu sai.");

    }

  };

  return (

    <View style={styles.container}>

      <TextInput

        style={styles.input}

        placeholder="Nhập mật khẩu"

        secureTextEntry

        value={password}

        onChangeText={setPassword}

      />

      <Button title="Đăng ký" onPress={handleRegister} />

      <Button title="Đăng nhập" onPress={handleLogin} />

      {storedHash ? (

        <Text style={styles.hash}>Giá trị băm: {storedHash}</Text>

      ) : null}

    </View>

  );

};

**Chạy thử**

A screenshot of a phone

Description automatically generated

Hình 11: Mẫu chạy thử SHA256 Encoding trên React Native 1.

A screenshot of a chat

Description automatically generated

Hình 12: Mẫu chạy thử SHA256 Encoding trên React Native 2.

# [PROFANITY FILTER](https://aboutreact.com/react-native-profanity-filter)

**Tổng quan:** là một công cụ hoặc hệ thống được sử dụng để phát hiện và loại bỏ các từ ngữ thô tục, không phù hợp hoặc xúc phạm trong văn bản. Các bộ lọc này thường được ứng dụng trong nhiều lĩnh vực như trò chuyện trực tuyến, mạng xã hội, game, và các ứng dụng nội dung người dùng để đảm bảo môi trường an toàn và lành mạnh cho tất cả mọi người.

**Đặc điểm:**

* Bảo vệ Người Dùng: Ngăn chặn sự xuất hiện của ngôn ngữ thô tục có thể gây tổn thương cho người khác.
* Tạo Môi Trường Tích Cực: Giúp duy trì một không gian trực tuyến thân thiện, dễ chịu và chuyên nghiệp.
* Tuân Thủ Quy định: Đáp ứng yêu cầu của các quy định pháp luật và chính sách nội bộ về nội dung.

**Cách cài đặt**

Cài đặt thư viện “bad-words”

npm install bad-words

Import component Filter từ thư viện bad-word

// Profinity Filter in React Native - To Remove bad words

// https://aboutreact.com/react-native-profanity-filter/

// Import React

import React, { useState } from 'react';

// Import all the required components

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

// Import profanity filter

import Filter from 'bad-words';

const App = () => {

  let [inputValue, setInputValue] = useState('');

  const filter = new Filter();

  const handleInput = (value) => {

    setInputValue(value);

  };

  return (

    <SafeAreaView style={{ flex: 1 }}>

      <View style={styles.container}>

        <Text style={styles.heading}>

          Profinity Filter in React Native {'\n'} To Remove Bad Words

        </Text>

        <Text style={styles.insertedTextStyle}>

          {inputValue ? filter.clean(inputValue) : ''}

        </Text>

        <TextInput

          value={inputValue}

          onChangeText={handleInput}

          placeholder={'Pleas Enter any Value'}

          style={styles.inputStyle}

        />

        <Text style={styles.textStyle}>

          Please insert any string with bad word in input and you will see it

          filtered out

        </Text>

      </View>

    </SafeAreaView>

  );

};

const styles = StyleSheet.create({

  container: {

    flex: 1,

    alignItems: 'center',

    backgroundColor: '#ffffff',

    padding: 16,

  },

  heading: {

    fontSize: 22,

    textAlign: 'center',

    marginBottom: 16,

    marginTop: 150,

  },

  textStyle: { marginVertical: 10, textAlign: 'center' },

  insertedTextStyle: {

    fontSize: 16,

    fontWeight: 'bold',

    marginVertical: 20,

    textAlign: 'center',

  },

  inputStyle: {

    width: '100%',

    height: 44,

    padding: 10,

    marginBottom: 10,

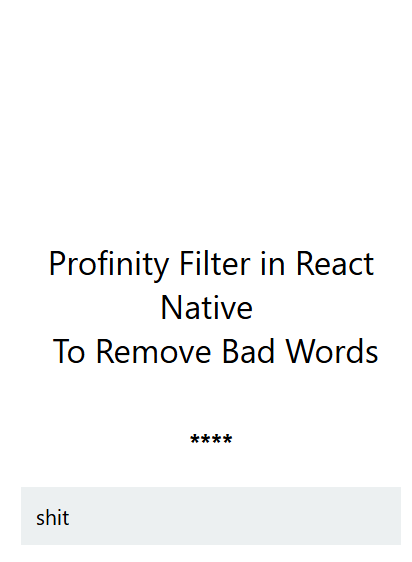
    backgroundColor: '#ecf0f1',

  },

});

export default App;

**Chạy thử**



Hình 13: Mẫu chạy thử IV. Profanity Filter trên React Native.

# KẾT LUẬN

Nội dung bài báo cáo seminar đã đạt được những kết quả sau:

Đã tìm hiểu, nghiên cứu về các tính năng cơ bản như Chart and graphs,hyperlinking, encoding. Những tính năng này rất quan trọng để tạo ra những ứng dụng không chỉ hoạt động tốt mà còn mang đến trải nghiệm người dùng thú vị và dễ sử dụng.

Có đánh giá sau thực nghiệm.

Tuy nhiên, bài Seminar vẫn còn một số hạn chế sau:

Do năng lực của bản thân và thời gian có hạn nên chưa thể tìm hiểu sâu về các tính năng trên để nâng cao trải nghiệm người dùng với các hiệu ứng chuyển động và bảo mật đa ngôn ngữ và chuẩn hóa UX/UI đa văn hóa.

**Hướng phát triển**

Nhóm sinh viên sẽ tiếp tục nghiên cứu sâu hơn về các tính năng trên nhằm giúp ích cho công tác sau này khi trải nghiệm thực tế tại đơn vị công tác.

# TÀI LIỆU THAM KHẢO

https://aboutreact.com/

|  |  |
| --- | --- |
|  |  |