**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 APP INTRO SLIDER, SPLASH SCREENS, BLUR & GRADIENT COMPONENTS**

**GVHD: ThS. Võ Ngọc Tân**

**Nhóm sinh viên thực hiện**:

1. **Nguyễn Thị Thanh Thảo 22540022**
2. **Trương Minh Trí 22540024**

**🙢 Tp. Hồ Chí Minh, 10/2024 🙠**

# NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

*……., ngày……...tháng……năm 20…*

**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** |
| Nguyễn Thị Thanh Thảo | 22540022 | Tìm hiểu App Slider | Hoàn thành tốt |
| Trương Minh Trí | 22540024 | Tìm hiểu Splash Screen, Blur & Gradien Components | Hoàn thành tốt |

**LỜI MỞ ĐẦU**

Trong thiết kế ứng dụng di động hiện đại, việc tạo trải nghiệm người dùng hấp dẫn ngay từ ban đầu là yếu tố quan trọng giúp giữ chân người dùng và tăng mức độ tương tác, bằng cách sử dụng các thành phần như App intro sliders và Splash screens.

Intro slider là các màn hình giới thiệu có thể điều hướng qua lại, thường xuất hiện khi người dùng lần đầu mở ứng dụng, để khám phá các tính năng chính, giá trị cốt lõi của ứng dụng bằng cách trực quan và sinh động.

Splash screen là màn hình chờ hiển thị logo hoặc hình ảnh đại diện của ứng dụng khi khởi động. Ngoài tăng tính nhận diện thương hiệu, Splash screen còn có thể ẩn đi quá trình tải dữ liệu ban đầu khi người dùng khởi chạy, mang lại cảm giác chuyên nghiệp và mượt mà trong trải nghiệm của người dùng.

Bên cạnh đó, hiệu ứng Blur (làm mờ) và Gradient (chuyển màu) là công cụ thiết kế quan trọng giúp tăng trải nghiệm thị giác. Hiệu ứng Blur thường dùng để tạo lớp nền mờ phía sau các thành phần chính, làm tăng sự nổi bật nội dung hoặc tạo chiều sâu cho giao diện. Đây là một phương pháp tinh tế để nhấn mạnh các yếu tố nội dung, đồng thời tạo cảm giác mềm mại và nhẹ nhàng cho người dùng khi điều hướng ứng dụng. Gradient lại là một kỹ thuật chuyển đổi màu sắc, giúp ứng dụng trông hiện đại và thu hút. Với khả năng tạo các dải màu chuyển tiếp mượt mà, Gradient mang đến cảm giác năng động, sống động và tạo ra dấu ấn cá nhân cho giao diện.

Bài viết này, nhóm sinh viên sẽ trình bày chi tiết các nội dung sau:

* Chương 1: Tìm hiểu về App intro slider

Màn hình giới thiệu nhanh ban đầu các tính năng hoặc hướng dẫn chính của ứng dụng thông qua các silde đầu tiên khi người dùng mở ứng dụng.

* Chương 2: Tìm hiểu về Splash Screens

Màn hình giới thiệu ngắn hiển thị logo và thương hiệu khi ứng dụng khởi động

* Chương 3: Tìm hiểu về Blur & Gradient Components

Blur và Gradient tạo hiệu ứng nền mờ và chuyển màu, giúp giao diện ứng dụng trở nên tinh tế và thu hút hơn.

* Chương 4: Kết luận

Tổng kết các kết quả đạt được sau khi tìm hiểu các nội dung đã đề xuất, rút ra những nội dung đã đạt được và chưa đạt được.

# MỤC LỤC

[NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN 2](#_Toc181439158)

[BẢNG PHÂN CÔNG, ĐÁNH GIÁ THÀNH VIÊN 3](#_Toc181439159)

[LỜI MỞ ĐẦU 4](#_Toc181439160)

[MỤC LỤC 5](#_Toc181439161)

[Chương 1. APP INTRO SLIDER 6](#_Toc181439162)

[1.1. View Pager with 3 Indicators 6](#_Toc181439163)

[1.2. App Intro Slider 11](#_Toc181439164)

[1.3. Step by Step Walkthrough 20](#_Toc181439165)

[Chương 2. Tìm hiểu về Splash Screens 30](#_Toc181439166)

[2.1. Định nghĩa 30](#_Toc181439167)

[2.2. Mục đích: 30](#_Toc181439168)

[2.3. Thành phần của Splash Screen: 30](#_Toc181439169)

[2.4. Phân loại Splash Screen: gồm hai loại Splash Screen phổ biến: 30](#_Toc181439170)

[2.5. Animated Splash Screen trong React Native 30](#_Toc181439171)

[2.6. Animation Splash Screen with Zoom Effect 31](#_Toc181439172)

[2.7. Lợi ích: 32](#_Toc181439173)

[Chương 3. Tìm Hiểu Về Blur & Gradient Components 33](#_Toc181439174)

[3.1. Blur Component 33](#_Toc181439175)

[3.2. Linear Gradient Component 35](#_Toc181439176)

[Chương 4. KẾT LUẬN 37](#_Toc181439177)

[TÀI LIỆU THAM KHẢO 38](#_Toc181439178)

# DANH MỤC HÌNH ẢNH

[Hình 1.1. Demo View Pager 9](#_Toc181448718)

[Hình 1.2. Demo Intro Slider đơn giản (1) 15](#_Toc181448719)

[Hình 1.3. Demo Intro Slider đơn giản (2) 16](#_Toc181448720)

[Hình 1.4. Màn hình kết thúc của Intro Slider 17](#_Toc181448721)

[Hình 1.5. Demo Intro Slider có Button ở giữa (1) 18](#_Toc181448722)

[Hình 1.6. Demo Intro Slider có Button ở giữa (2) 19](#_Toc181448723)

[Hình 1.7. Demo Intro Slider với Button tùy chỉnh (1) 20](#_Toc181448724)

[Hình 1.8. Demo Intro Slider với Button tùy chỉnh (2) 21](#_Toc181448725)

[Hình 1.9. Demo Step by Step Walkthrough sử dụng react-native-copilot 24](#_Toc181448726)

[Hình 1.10. Demo Step by Step Walkthrough sử dụng các thành phần React Native sẵn có (1) 25](#_Toc181448727)

[Hình 1.11. Demo Step by Step Walkthrough sử dụng các thành phần React Native sẵn có (2) 26](#_Toc181448728)

[Hình 2.1. Màn hình hiển thị hiệu ứng Animated Splash Screen 32](#_Toc181448729)

[Hình 2.2. Màn hình hiển thị hiệu ứng Animated Splash Screen with Zoom Effect 33](#_Toc181448730)

[Hình 3.1. Màn hình ứng dụng trước khi áp dụng hiệu ứng làm mờ 34](#_Toc181448731)

[Hình 3.2. Xlight 35](#_Toc181448732)

[Hình 3.3. Light 35](#_Toc181448733)

[Hình 3.4. Dark 35](#_Toc181448734)

[Hình 3.5. Code Simple Gradient 36](#_Toc181448735)

[Hình 3.6. Code Horizontal Gradient 36](#_Toc181448736)

[Hình 3.7. Code Location Gradient 37](#_Toc181448737)

[Hình 3.8. Màn hình hiển thị hiệu ứng Linear Gradient 38](#_Toc181448738)

# APP INTRO SLIDER

## View Pager with 3 Indicators

View Pager là một thành phần giao diện người dùng trong ứng dụng di động cho phép người dùng chuyển đổi giữa các trang nội dung bằng cách vuốt sang trái hoặc phải. Thành phần này thường được sử dụng để hiển thị nội dung theo dạng trang (page-based content), như các hình ảnh, bài viết.

Để sử dụng View Pager, cần cài đặt **@shankarmorwal/rn-viewpager** thông qua câu lệnh:

npm install @shankarmorwal/rn-viewpager –save

Thư viện **@shankarmorwal/rn-viewpager –save** có tích hợp sẵncác tính năng sau:

* **Dot Indicator**: Thư viện này cung cấp các chỉ báo dot có thể tùy chỉnh để hiển thị trạng thái của các trang hiện tại.
* **Tab Indicator**: Bạn có thể sử dụng tab để điều hướng giữa các trang một cách dễ dàng.
* **Title Indicator**: Thư viện này cũng cho phép bạn hiển thị tiêu đề cho từng trang.

Tuy nhiên, do nhóm demo trên Expo Go, mà thư viện này không được hỗ trợ trực tiếp trong môi trường Expo Managed Workflow nên nhóm sử dụng thư viện **react-native-pager-view** được tích hợp sẵn trong Expo, cài đặt thông qua câu lệnh:

npx expo install react-native-pager-view

Thư viện này có một nhược điểm là những chỉ báo trên không được tích hợp sẵn, nếu muốn sử dụng các chỉ báo này trong ứng dụng cần tự xây dựng chúng.

### Cấu trúc của View Pager

* PagerView: là thành phần chính cho việc tạo View Pager trong React Native. Nó cho phép hiển thị nhiều trang và quản lý chuyển động giữa chúng.
* Nội dung Trang: chứa các thành phần như văn bản, hình ảnh hoặc biểu mẫu.

### Các chỉ báo (Indicators)

Chỉ báo giúp người dùng nhận biết được họ đang ở trang nào trong View Pager. Dưới đây là một số loại chỉ báo phổ biến:

* **Dot Indicator (chỉ báo dấu chấm):**
* Chức năng: Hiển thị một hàng các dấu chấm, mỗi dấu chấm tương ứng với một trang. Dấu chấm được tô đậm hoặc đổi màu khi người dùng vuốt tới trang đó.
* Sử dụng: Giúp người dùng dễ dàng xác định số lượng trang và vị trí hiện tại của họ trong View Pager.
* **Title Indicator (chỉ báo tiêu đề):**
* Chức năng: Hiển thị tiêu đề của từng trang bên dưới PagerView. Tiêu đề này có thể đổi màu hoặc được in đậm khi trang đó được chọn.
* Sử dụng: Cung cấp thông tin rõ ràng về nội dung của trang đang hiển thị.
* **Tab Indicator (chỉ báo thẻ):**
* Chức năng: Cung cấp các thẻ (tabs) để người dùng có thể nhấn để chuyển trực tiếp đến trang tương ứng mà không cần phải vuốt.
* Sử dụng: Thường được đặt ở phía dưới cùng của màn hình, cho phép người dùng dễ dàng truy cập vào các trang cụ thể.

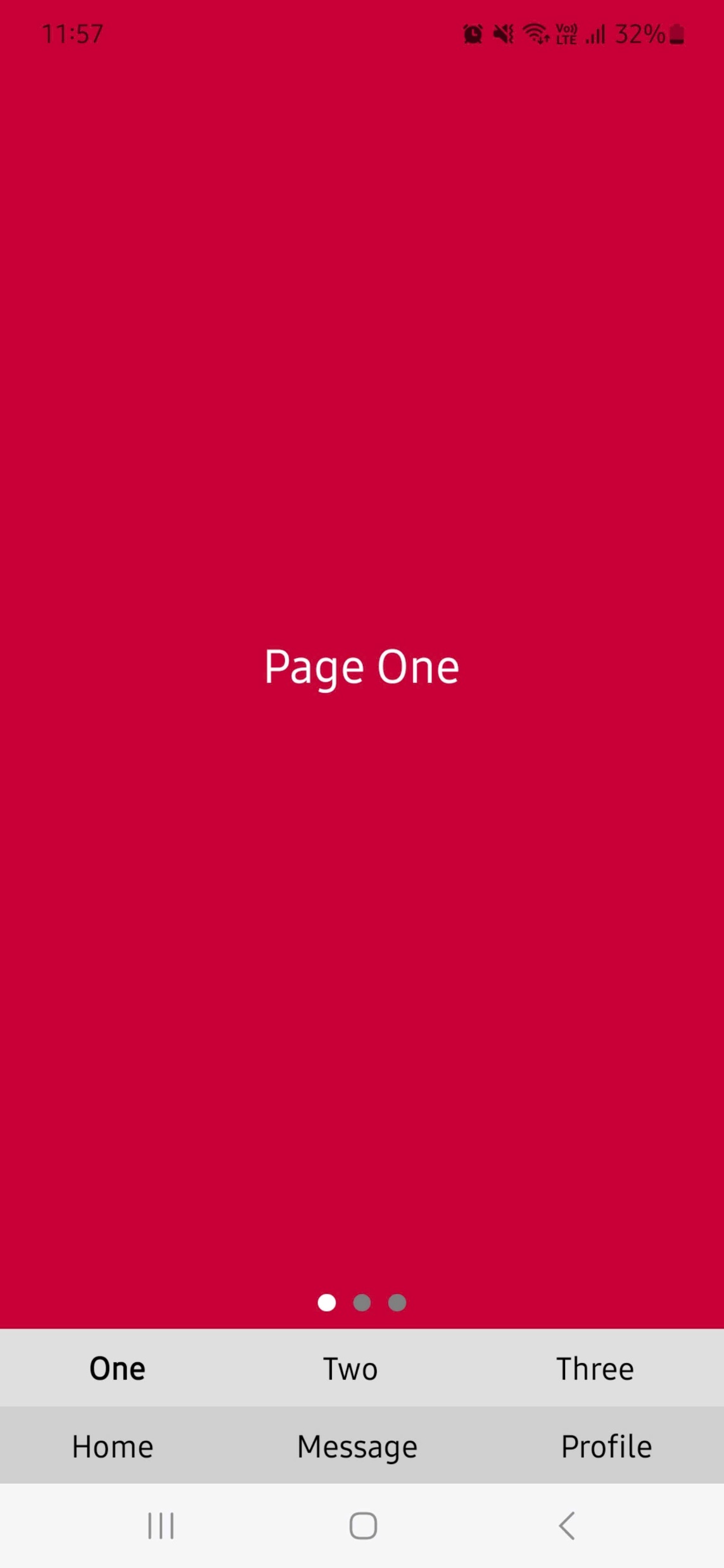
### Cách hoạt động của View Pager và Indicators

* Khi người dùng vuốt sang trái hoặc phải, View Pager sẽ chuyển sang trang tiếp theo hoặc trước đó.
* Trong quá trình chuyển đổi, Dot Indicator sẽ cập nhật để phản ánh trang hiện tại bằng cách thay đổi màu hoặc hình dạng của dấu chấm.
* Title Indicator sẽ hiển thị tiêu đề của trang hiện tại, giúp người dùng nhận diện nội dung một cách nhanh chóng.
* Nếu có Tab Indicator, người dùng có thể nhấn vào các thẻ để chuyển đến trang cụ thể mà không cần phải vuốt.

### Lợi ích của View Pager và Indicators

* Cải thiện trải nghiệm người dùng: Giúp người dùng dễ dàng điều hướng giữa các trang nội dung.
* Tăng cường tính tương tác: Cho phép người dùng tương tác nhanh chóng với nội dung.
* Thông tin rõ ràng: Các chỉ báo giúp cung cấp thông tin trực quan về vị trí và nội dung trang.

### Hình ảnh demo



Hình 1.1. Demo View Pager

### Source code:

* Link source code gốc: <https://aboutreact.com/react-native-view-pager/>
* Source code được điều chỉnh sử dụng thư viện tương thích với Expo:

import React, { useState } from 'react';

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

import PagerView from 'react-native-pager-view';

*// Custom Dot Indicator component positioned at the bottom of each page*

**const** DotIndicator = ({ pageCount, selectedIndex }) **=>** (

  <View style={styles.dotContainer}>

    {[...Array(pageCount)].map((\_, index) **=>** (

      <View

        key={index}

        style={[

          styles.dot,

          selectedIndex === index && styles.selectedDot,

        ]}

      />

    ))}

  </View>

);

*// Custom Title Indicator at the top of the PagerView*

**const** TitleIndicator = ({ titles, currentIndex }) **=>** (

  <View style={styles.titleContainer}>

    {titles.map((title, index) **=>** (

      <Text

        key={index}

        style={[

          styles.titleText,

          currentIndex === index && styles.boldTitle,

        ]}

      >

        {title}

      </Text>

    ))}

  </View>

);

*// Custom Tab Indicator component at the bottom*

**const** TabIndicator = ({ tabs, setCurrentPage }) **=>** (

  <View style={styles.tabContainer}>

    {tabs.map((tab, index) **=>** (

      <TouchableOpacity key={index} onPress={() **=>** setCurrentPage(index)}>

        <Text style={styles.tabText}>{tab}</Text>

      </TouchableOpacity>

    ))}

  </View>

);

export default **function** App() {

**const** [currentPage, setCurrentPage] = useState(0);

**const** titles = ['One', 'Two', 'Three'];

**const** tabs = ['Home', 'Message', 'Profile'];

**const** handlePageChange = (e) **=>** {

    setCurrentPage(e.nativeEvent.position);

  };

  return (

    <View style={styles.container}>

      {*/\* PagerView for main content \*/*}

      <PagerView

        style={styles.pagerView}

        initialPage={0}

        onPageSelected={handlePageChange}

      >

        <View style={[styles.page, styles.pageOne]} key="1">

          <Text style={styles.pageText}>Page One</Text>

          <DotIndicator pageCount={3} selectedIndex={currentPage} />

        </View>

        <View style={[styles.page, styles.pageTwo]} key="2">

          <Text style={styles.pageText}>Page Two</Text>

          <DotIndicator pageCount={3} selectedIndex={currentPage} />

        </View>

        <View style={[styles.page, styles.pageThree]} key="3">

          <Text style={styles.pageText}>Page Three</Text>

          <DotIndicator pageCount={3} selectedIndex={currentPage} />

        </View>

      </PagerView>

      {*/\* Title Indicator below the PagerView \*/*}

      <TitleIndicator titles={titles} currentIndex={currentPage} />

      {*/\* Tab Indicator at the bottom \*/*}

      <TabIndicator tabs={tabs} setCurrentPage={setCurrentPage} />

    </View>

  );

}

**const** styles = StyleSheet.create({

  container: {

    flex: 1,

    backgroundColor: '#F5F5F5',

  },

  pagerView: {

    flex: 1,

  },

  page: {

    flex: 1,

    justifyContent: 'center',

    alignItems: 'center',

    position: 'relative',

  },

  pageText: {

    fontSize: 24,

    color: 'white',

  },

  pageOne: {

    backgroundColor: '#C70039',

  },

  pageTwo: {

    backgroundColor: '#FF5733',

  },

  pageThree: {

    backgroundColor: '#FFC300',

  },

*// Dot Indicator at the bottom of each page*

  dotContainer: {

    flexDirection: 'row',

    justifyContent: 'center',

    position: 'absolute',

    bottom: 10,

  },

  dot: {

    width: 10,

    height: 10,

    borderRadius: 5,

    backgroundColor: 'gray',

    marginHorizontal: 5,

  },

  selectedDot: {

    backgroundColor: 'white',

  },

*// Title Indicator below the PagerView*

  titleContainer: {

    flexDirection: 'row',

    justifyContent: 'space-around',

    paddingVertical: 10,

    backgroundColor: '#E0E0E0',

  },

  titleText: {

    fontSize: 16,

    color: 'black',

  },

  boldTitle: {

    fontWeight: 'bold',

  },

*// Tab Indicator at the bottom*

  tabContainer: {

    flexDirection: 'row',

    justifyContent: 'space-around',

    paddingVertical: 10,

    backgroundColor: '#D0D0D0',

  },

  tabText: {

    fontSize: 16,

    color: 'black',

  },

});

## App Intro Slider

App Intro Slider là một tính năng rất phổ biến trong các ứng dụng di động hiện nay, giúp người dùng mới hiểu rõ về ứng dụng của bạn và các tính năng quan trọng của nó trước khi bắt đầu sử dụng.

App Intro Slider là một chuỗi các màn hình chào mừng (slides) hiển thị khi người dùng mở ứng dụng lần đầu tiên. Những màn hình này thường cung cấp một cái nhìn tổng quan về ứng dụng, giải thích các chức năng chính và hướng dẫn người dùng cách bắt đầu.

**\* Thư viện cần cài đặt:**

npm install react-native-app-intro-slider –save

### Cấu trúc của App intro slider

* Header (Tiêu đề): thể hiện nội dung chính của màn hình, giúp cung cấp thông tin nhanh chóng về ứng dụng/màn hình.
* Content Area (Khu vực nội dung): là nơi chứa các thông tin, hình ảnh hoặc mô tả về các tính năng của ứng dụng. Mỗi trang của slider có thể hiển thị một tính năng khác nhau, đi kèm với hình ảnh minh họa. Content Area giúp người dùng hiểu rõ hơn về cách sử dụng ứng dụng và các lợi ích mà nó mang lại. Các hình ảnh/minh họa sinh động thường được sử dụng để giúp người dùng hình dung rõ ràng hơn về các tính năng của ứng dụng, đồng thời giúp tăng tính hấp dẫn trực quan cho ứng dụng và hỗ trợ người dùng dễ dàng ghi nhớ thông tin.
* Navigation Buttons (Nút điều hướng): các nút điều hướng như "Previous", "Next", "Skip", hoặc "Finish" cho phép người dùng điều hướng giữa các bước của intro slider. Nút "Skip" thường cho phép người dùng bỏ qua phần giới thiệu và đi thẳng vào ứng dụng.
* Progress Indicator (Thanh tiến trình): có thể được biểu diễn dưới dạng một thanh, hoặc các chấm, hoặc vòng tròn chỉ ra người dùng đang ở bước nào trong quá trình giới thiệu, giúp người dùng nhận biết được họ đã xem được bao nhiêu phần và còn lại bao nhiêu bước.
* Call to Action: một nút để bắt đầu sử dụng ứng dụng ngay sau khi hoàn thành phần giới thiệu. Khuyến khích người dùng thực hiện hành động, như đăng nhập, đăng ký hoặc bắt đầu sử dụng ứng dụng.

### Cách hoạt động của App intro slider

App Intro Slider thường được kích hoạt tự động khi người dùng lần đầu sử dụng ứng dụng. Người dùng có thể di chuyển giữa các bước của App Intro thông qua các nút Next hoặc Skip. Một số App Intro có thể tự động chuyển giữa các màn hình sau một khoảng thời gian nhất định để hướng dẫn người dùng mà không cần phải tương tác.

Trong khi người dùng di chuyển giữa các màn hình, một chỉ báo tiến độ (thường là thanh hoặc vòng tròn) giúp người dùng nhận biết họ đang ở bước nào và còn lại bao nhiêu bước.

Khi người dùng đến màn hình cuối cùng của App Intro Slider, họ sẽ thấy một nút Finish, hoặc Done, hoặc Get Started để bắt đầu sử dụng ứng dụng. Nếu người dùng chọn nút Skip ở bất kỳ bước nào, họ sẽ được chuyển thẳng đến màn hình chính của ứng dụng mà không cần hoàn thành phần giới thiệu. Trong màn hình chính, người dùng có thể nhấn nút Show Intro Slider again để hiển thị lại các slide giới thiệu

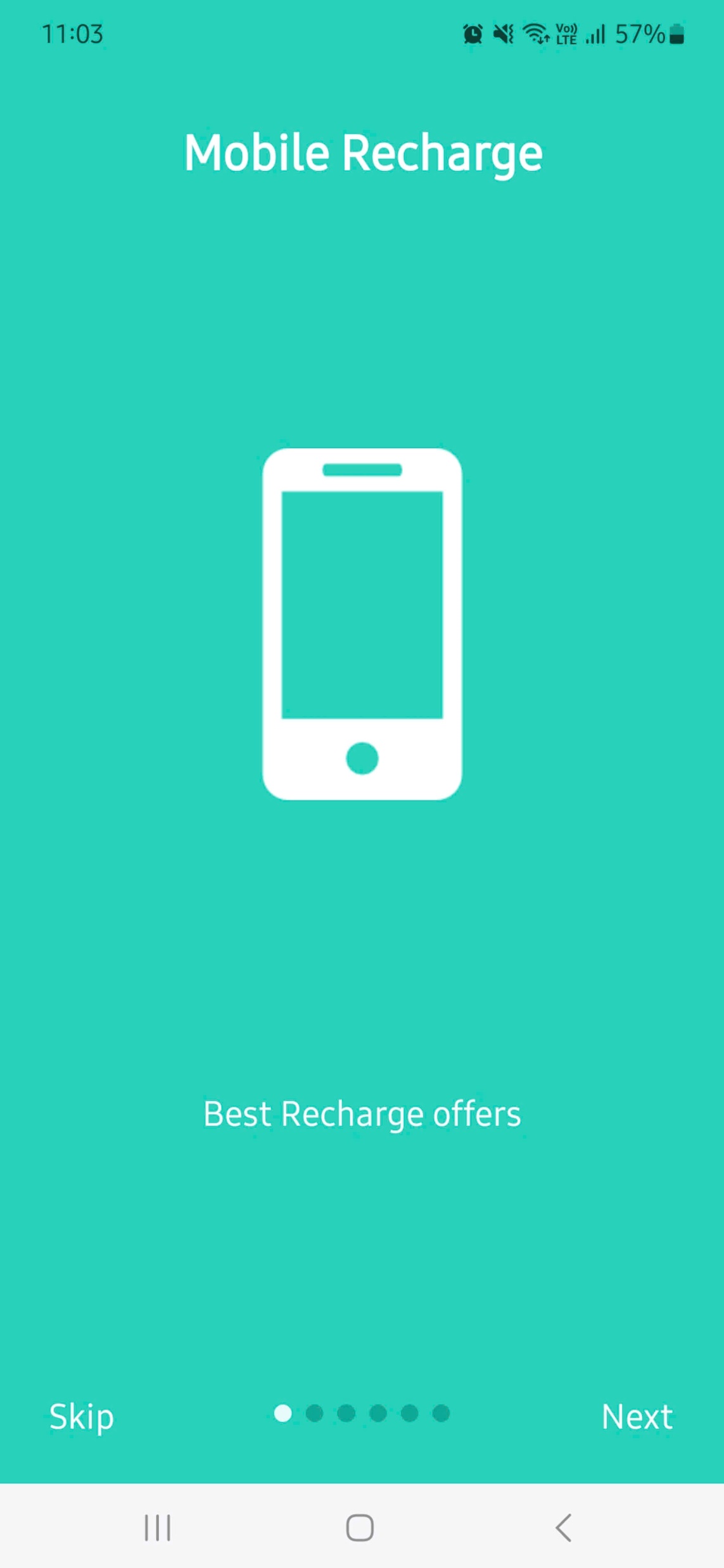
Đối với những lần khởi động tiếp theo của ứng dụng, App Intro Slider có thể được lập trình để không hiển thị nữa (sau khi người dùng đã xem nó một lần) bằng cách sử dụng các phương thức lưu trữ như AsyncStorage trên thiết bị di động.

### Lợi ích của App intro slider

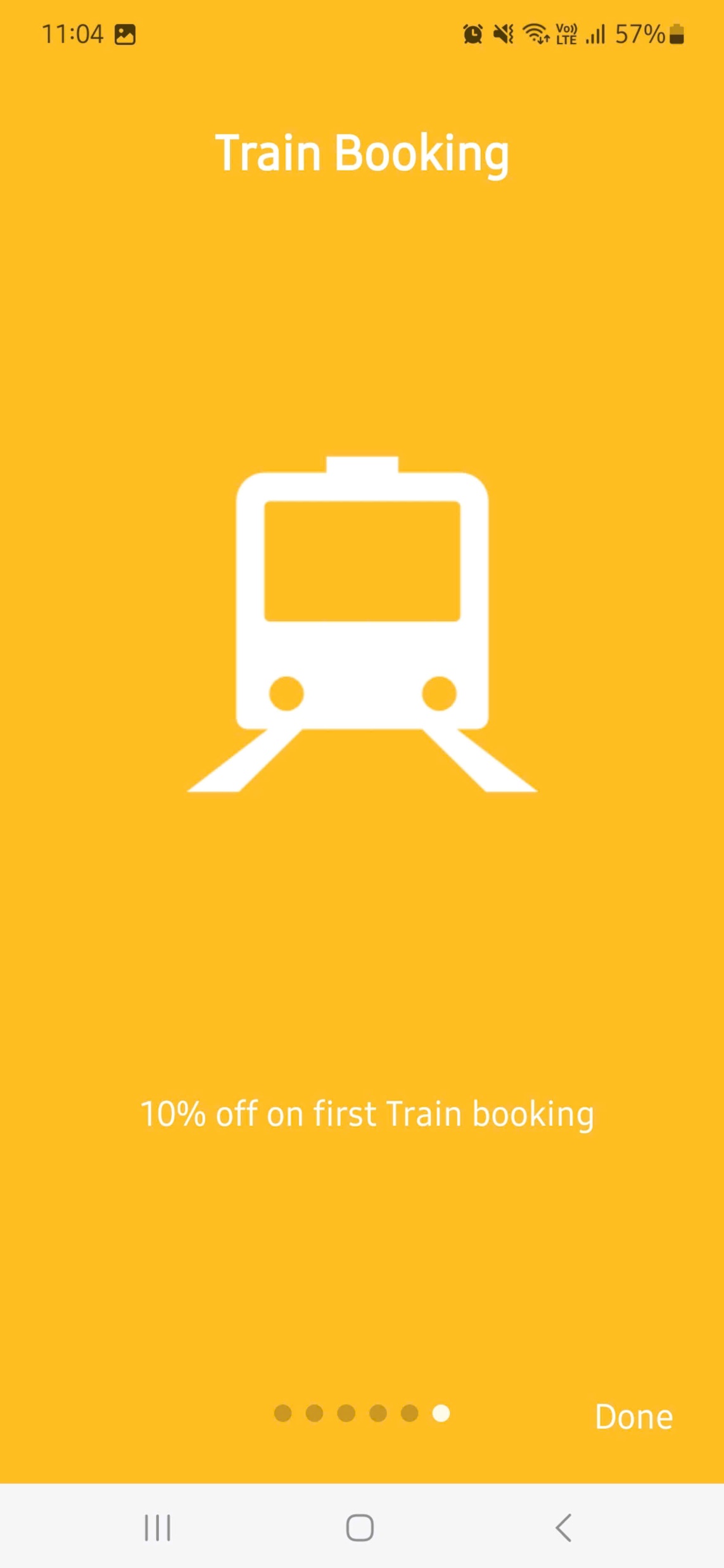
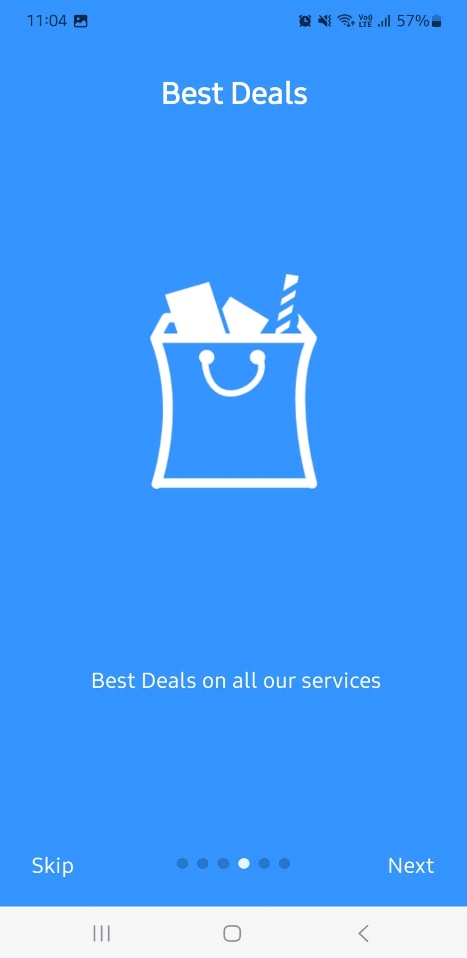
* Giúp người dùng làm quen với ứng dụng: cung cấp hướng dẫn cụ thể về cách sử dụng các tính năng của ứng dụng, giúp người dùng nhanh chóng làm quen mà không cảm thấy bối rối. Đồng thời giúp giảm thiểu sai sót và khó khăn trong quá trình sử dụng.
* Nâng cao trải nghiệm người dùng bằng cách tạo ấn tượng đầu tiên tốt về ứng dụng, tăng tỷ lệ giữ chân người dùng.
* Làm nổi bật các tính năng chính, tăng khả năng tương tác của người dùng.

### Hình ảnh demo

#### Intro slider đơn giản



Hình 1.2. Demo Intro Slider đơn giản (1)



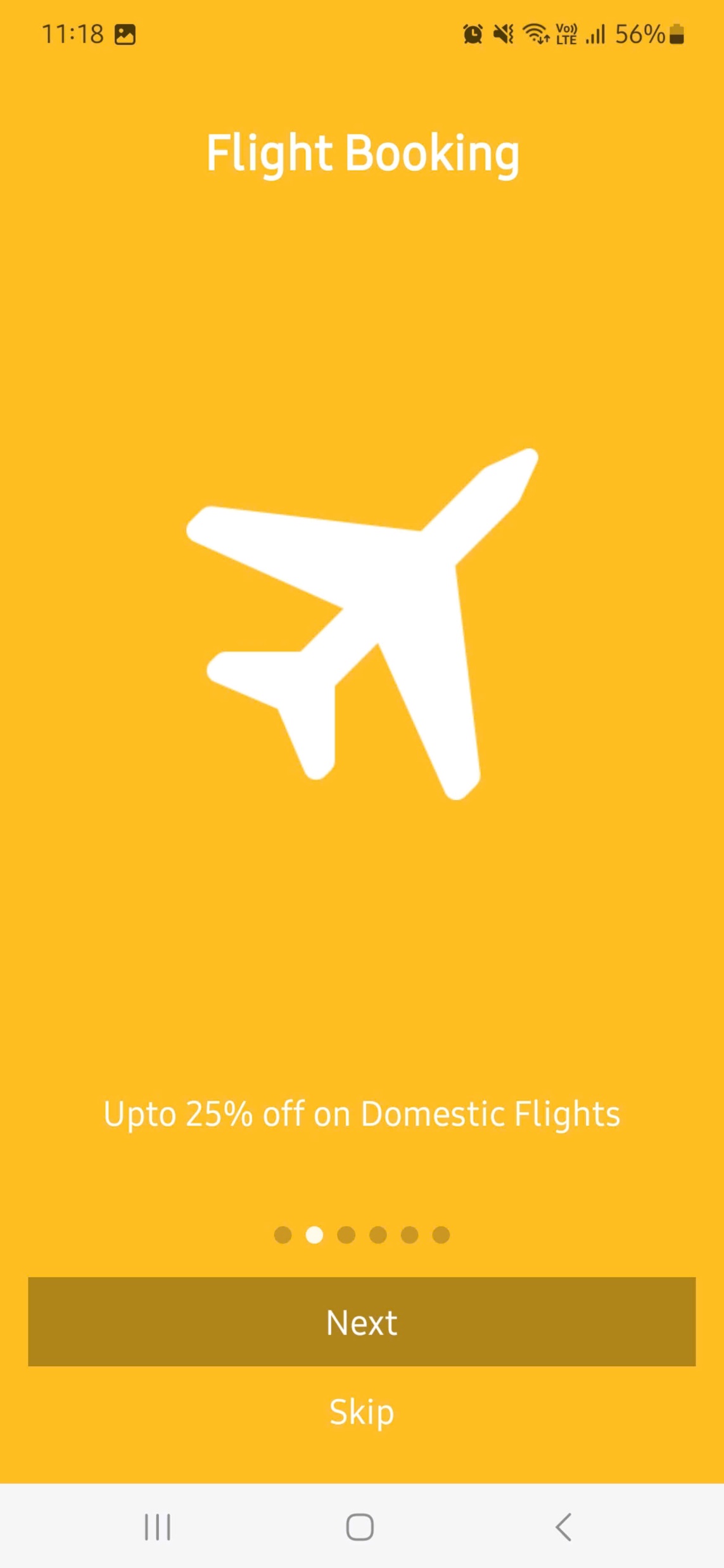
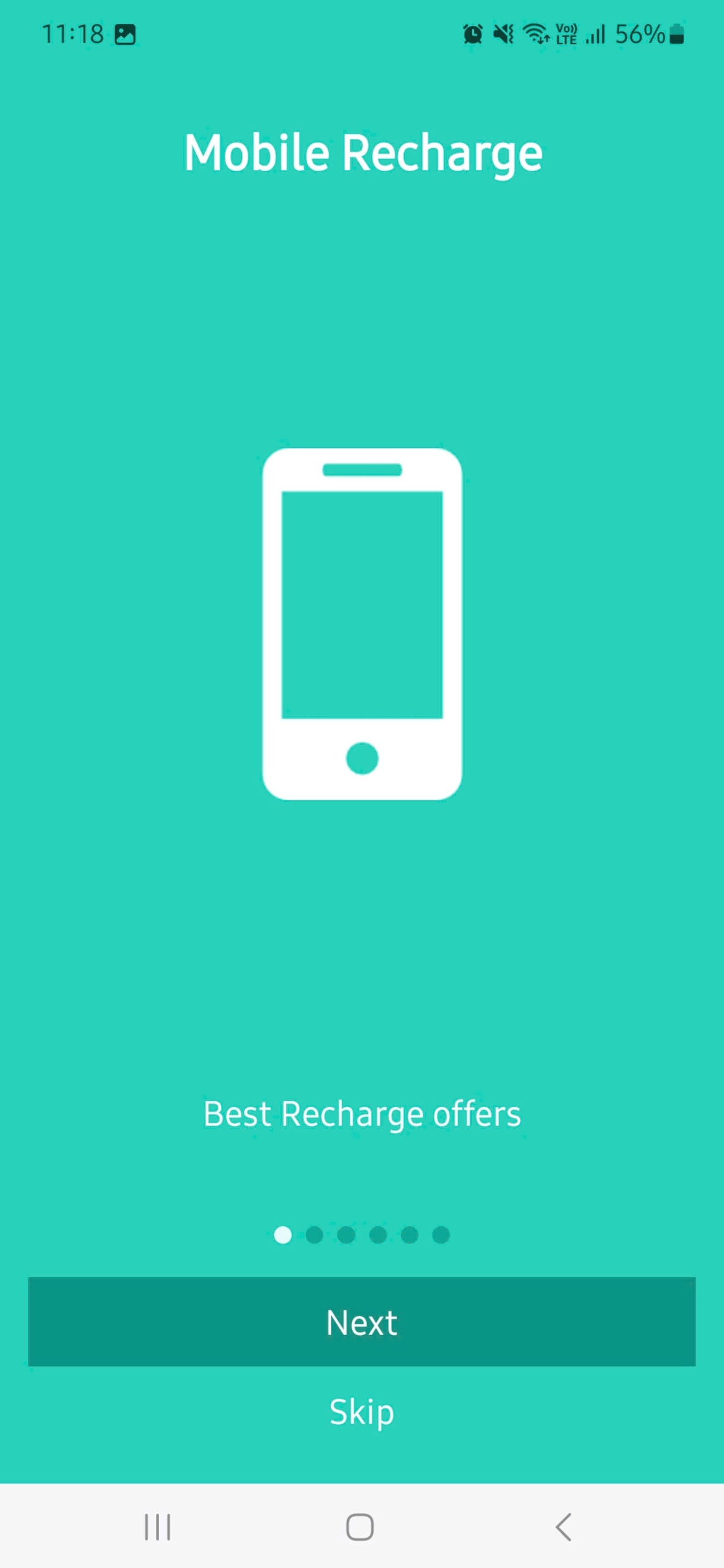
Hình 1.3. Demo Intro Slider đơn giản (2)

A screenshot of a slider

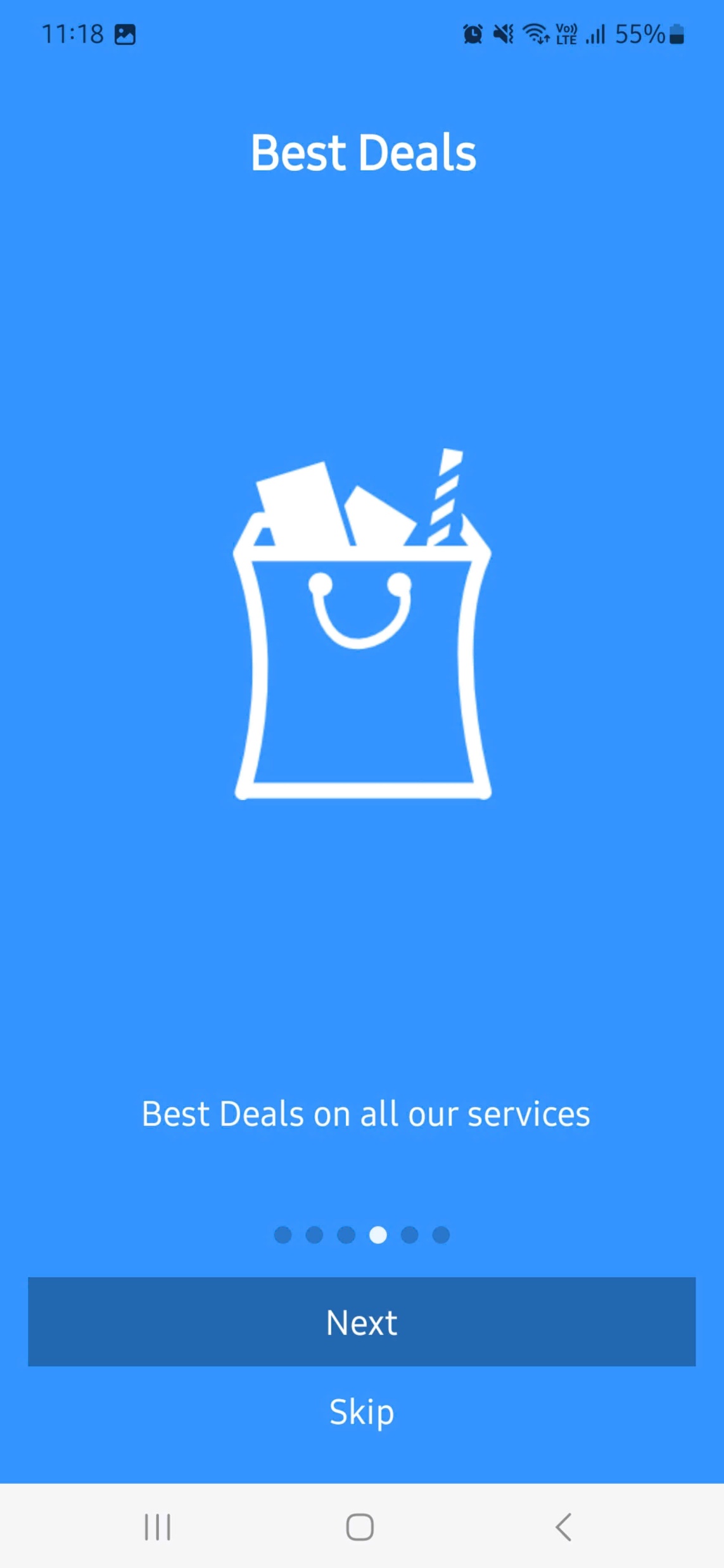
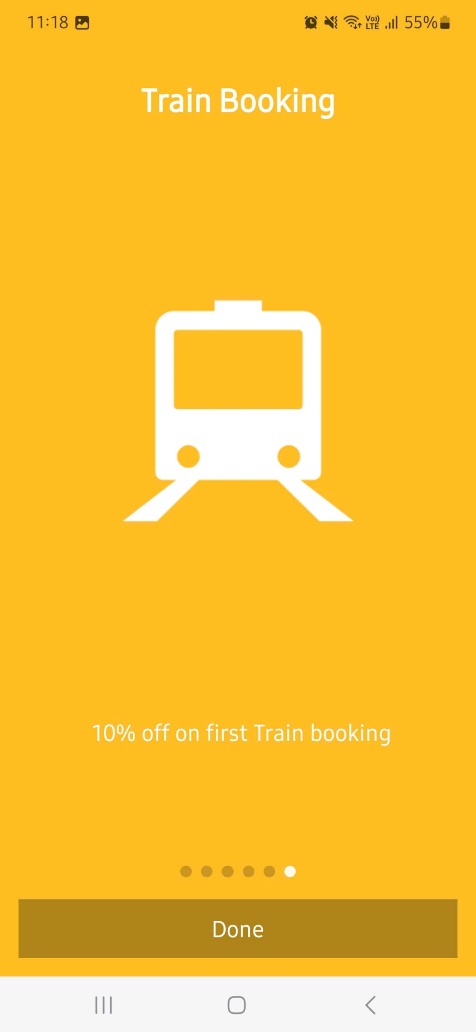
Description automatically generated

Hình 1.4. Màn hình kết thúc của Intro Slider

#### Intro slider có Button ở giữa

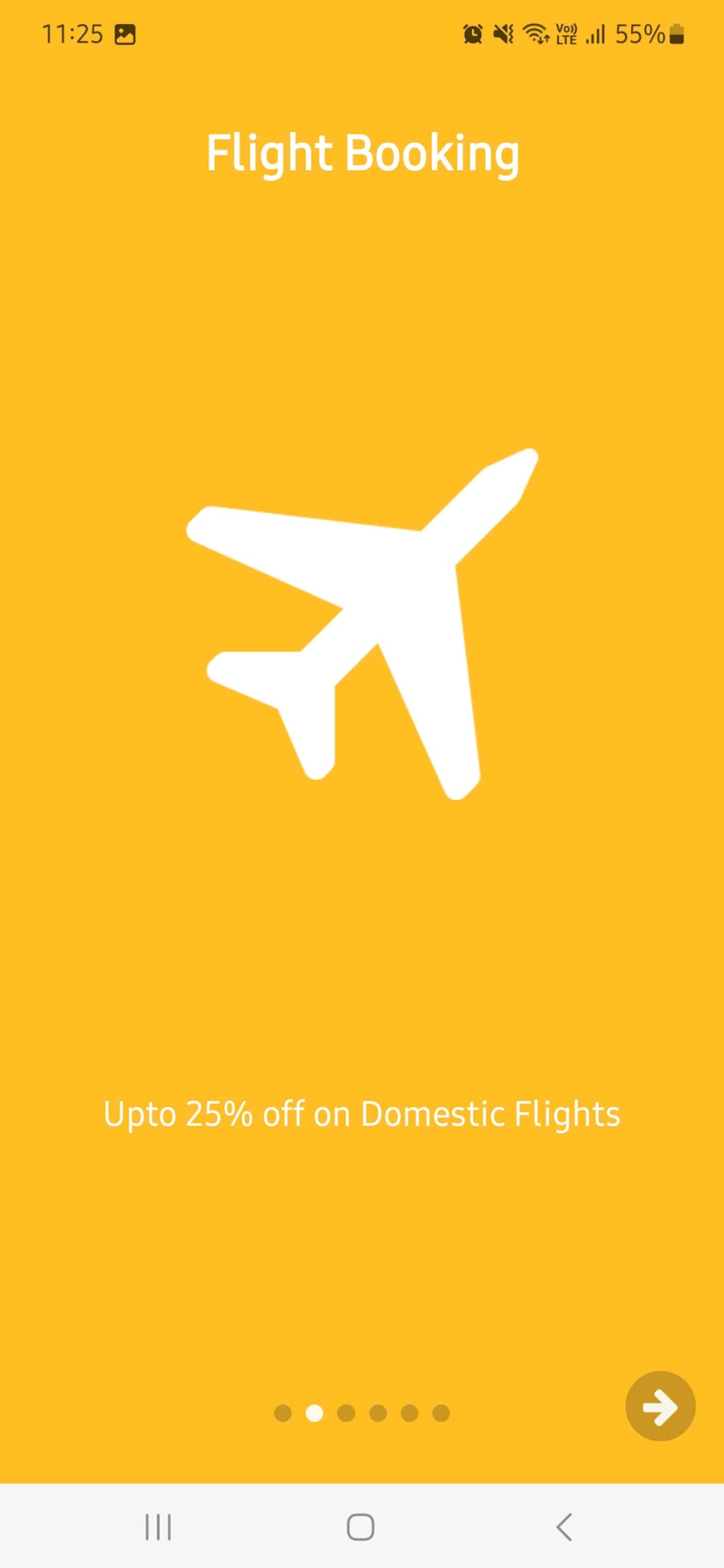
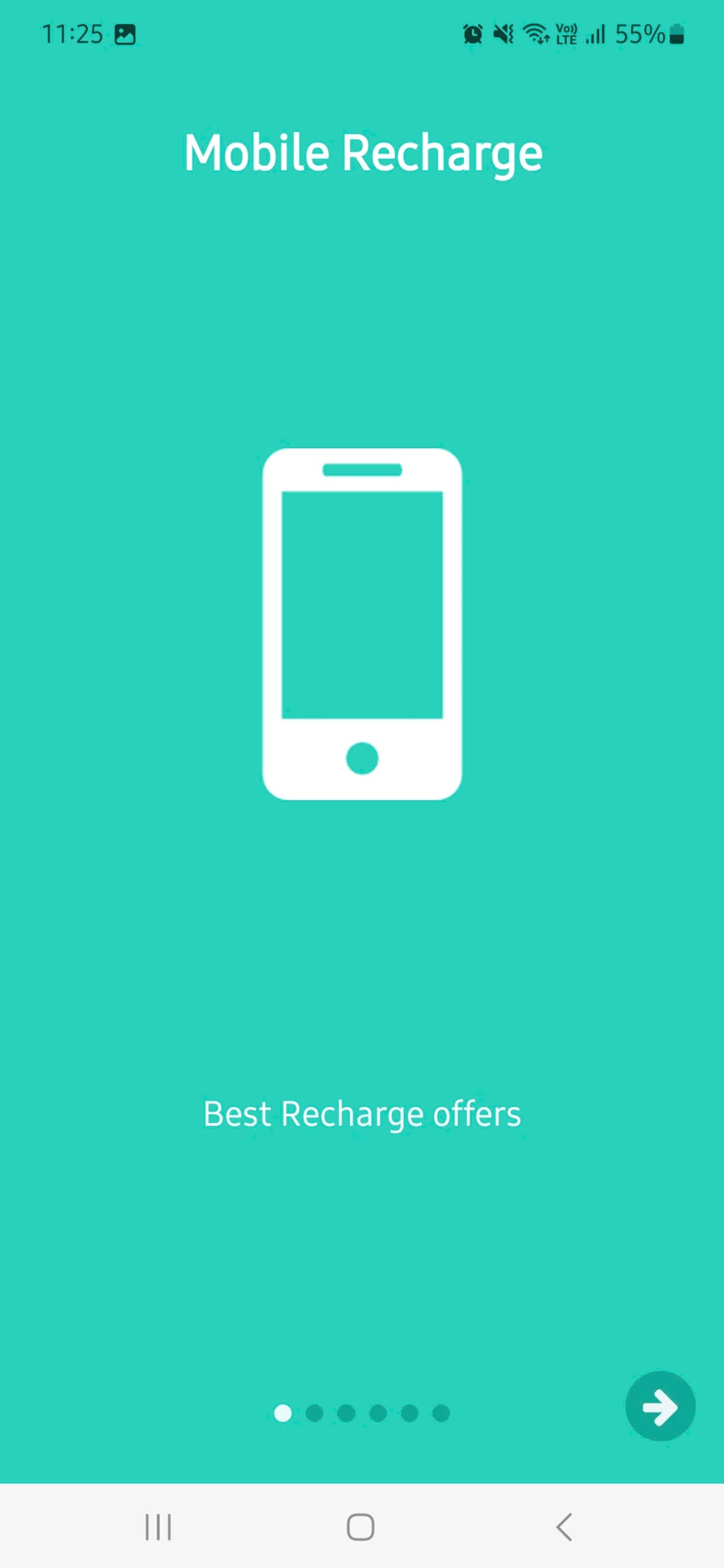
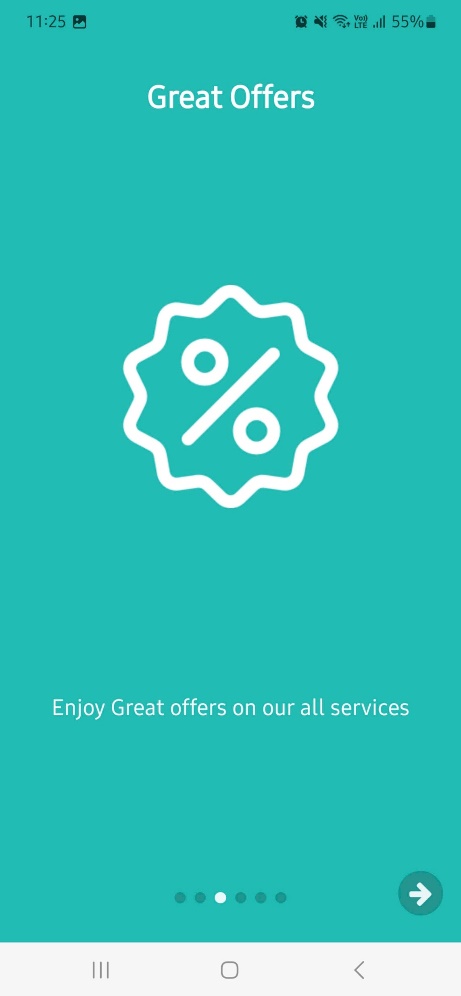


Hình 1.5. Demo Intro Slider có Button ở giữa (1)

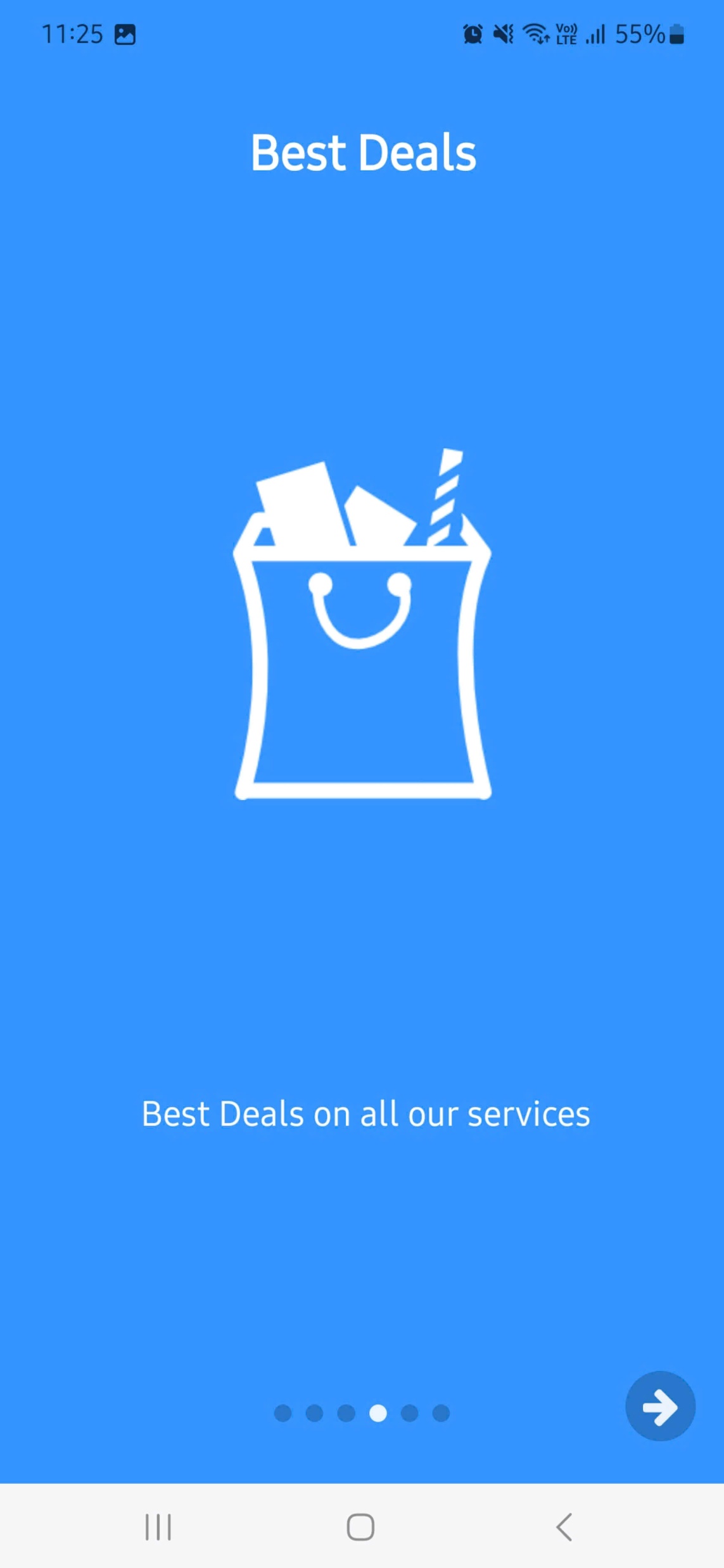
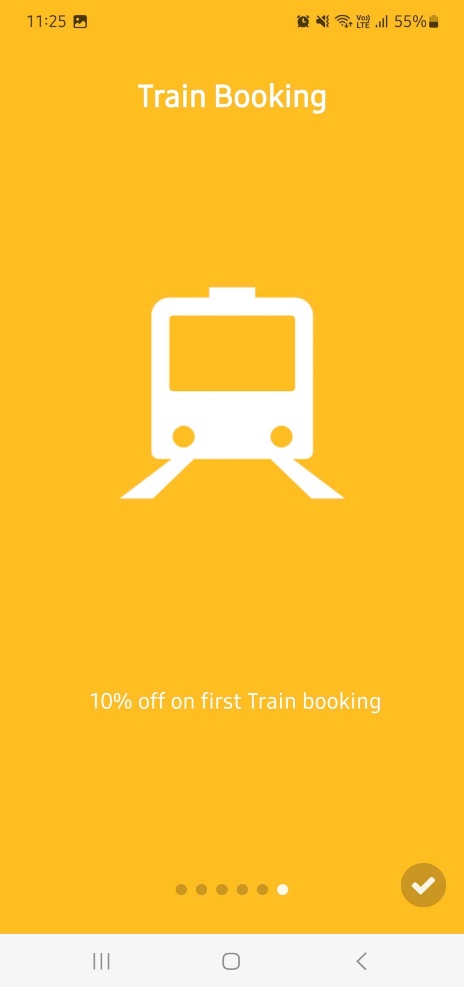


Hình 1.6. Demo Intro Slider có Button ở giữa (2)

#### Intro slider với Button tùy chỉnh



Hình 1.7. Demo Intro Slider với Button tùy chỉnh (1)



Hình 1.8. Demo Intro Slider với Button tùy chỉnh (2)

### Source code:

* Link source code gốc: <https://aboutreact.com/react-native-app-intro-slider/>

## Step by Step Walkthrough

Step by Step Walkthrough là một phương pháp hướng dẫn người dùng từng bước một trong quá trình trải nghiệm ứng dụng hoặc phần mềm. Đây là một trong những phương pháp hiệu quả nhất để giúp người dùng mới làm quen với các tính năng và cách sử dụng của ứng dụng một cách nhanh chóng và rõ ràng.

Step by Step Walkthrough tập trung vào việc dẫn dắt người dùng qua từng tính năng hoặc thao tác một cách chi tiết và trực quan. Trong các ứng dụng có nhiều tính năng hoặc quy trình phức tạp, phương pháp này giúp người dùng không bị choáng ngợp bởi toàn bộ hệ thống và có thể tập trung vào từng phần cụ thể, nhờ đó giúp người dùng nắm vững cách sử dụng các các tính năng chính.

Để tạo Step by Step Walkthrough, cần cài đặt thư viện **react-native-copilot** thông qua câu lệnh:

npm install --save react-native-copilot

Thư viện **react-native-copilot** hỗ trợ việc tạo các bước hướng dẫn tương tác trong ứng dụng với các tính năng như sau:

- **Highlight**: Làm nổi bật các phần tử trên màn hình mà người dùng cần chú ý.

- **Tooltip**: Hiển thị các thông tin bổ sung cho người dùng để giải thích chức năng của phần tử.

- **Navigation giữa các bước**: Cung cấp các bước điều hướng như Next, Previous, Finish để hướng dẫn người dùng dễ dàng.

- **Customizable Styling**: Cho phép tuỳ chỉnh giao diện của các tooltip, màu sắc và highlight để phù hợp với phong cách của ứng dụng.

Thư viện này giúp các nhà phát triển tạo walkthrough hiệu quả mà không cần phải viết lại nhiều mã từ đầu.

Nếu sử dụng trên Expo Go, thư viện **react-native-copilot** không được hỗ trợ trực tiếp. Nhóm đề xuất một số phương án thay thế:

- Sử dụng thư viện **react-native-walkthrough-tooltip**, cài đặt thông qua câu lệnh:

expo install react-native-walkthrough-tooltip

Đây là thư viện chuyên dụng để tạo các tooltip hướng dẫn ở từng bước. Nó hiển thị các hướng dẫn ở các vị trí được chỉ định, tương tự như các bước của **react-native-copilot**. Tuy nhiên, thư viện này không có tính năng quản lý chuỗi các bước (sequence of steps) tự động như Copilot, nên cần quản lý logic chuyển bước

- Chuyển sang dùng các thành phần React Native mặc định với cấu trúc tự tạo. Phương án này có ưu điểm là hoàn toàn tuỳ chỉnh, không phụ thuộc vào thư viện bên ngoài, dễ tích hợp với Expo mà không lo vấn đề tương thích. Tuy nhiên, sẽ cần nhiều công sức để quản lý các bước và hiệu ứng, cũng như logic điều hướng.

### Cấu trúc của Step by step walkthrough:

* Màn hình giới thiệu (Intro Screen): là màn hình đầu tiên mà người dùng nhìn thấy, thường bao gồm một tiêu đề và mô tả ngắn gọn về ứng dụng hoặc quy trình mà người dùng sẽ thực hiện.
* Các bước hướng dẫn (Step Screens): Mỗi bước trong hướng dẫn sẽ có một màn hình riêng biệt, hoặc sẽ được làm nổi bật lên.
* Nút điều hướng: các nút như Next, Previous và Skip giúp người dùng dễ dàng điều hướng qua các bước.
* Thông tin hỗ trợ (Tooltips): Các chú thích nhỏ có thể xuất hiện để cung cấp thêm thông tin về chức năng mà người dùng đang tìm hiểu.
* Màn hình cuối cùng thường cho người dùng biết rằng họ đã hoàn tất hướng dẫn. Có thể có một nút để đến trang chính của ứng dụng hoặc khởi động lại hướng dẫn.

### Cách hoạt động của Step by step walkthrough:

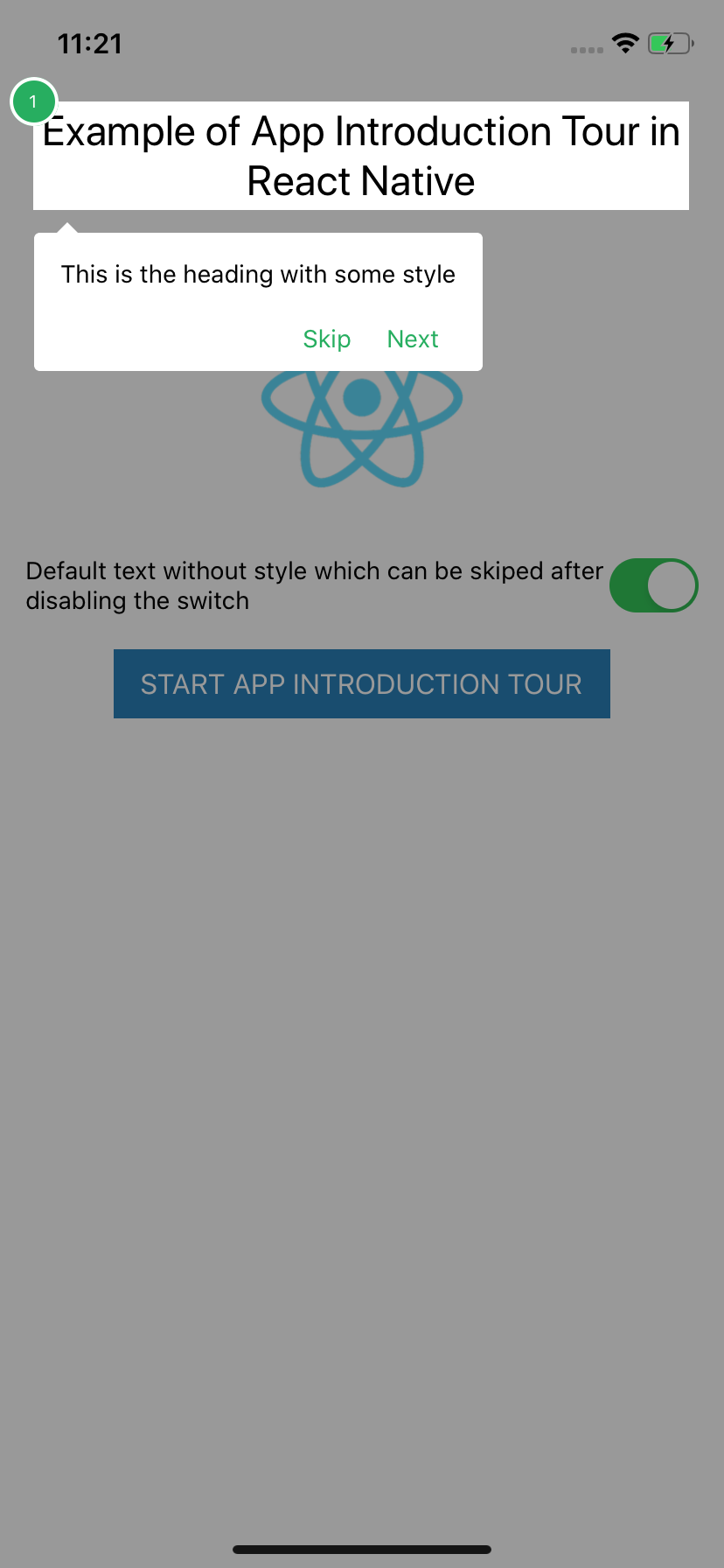
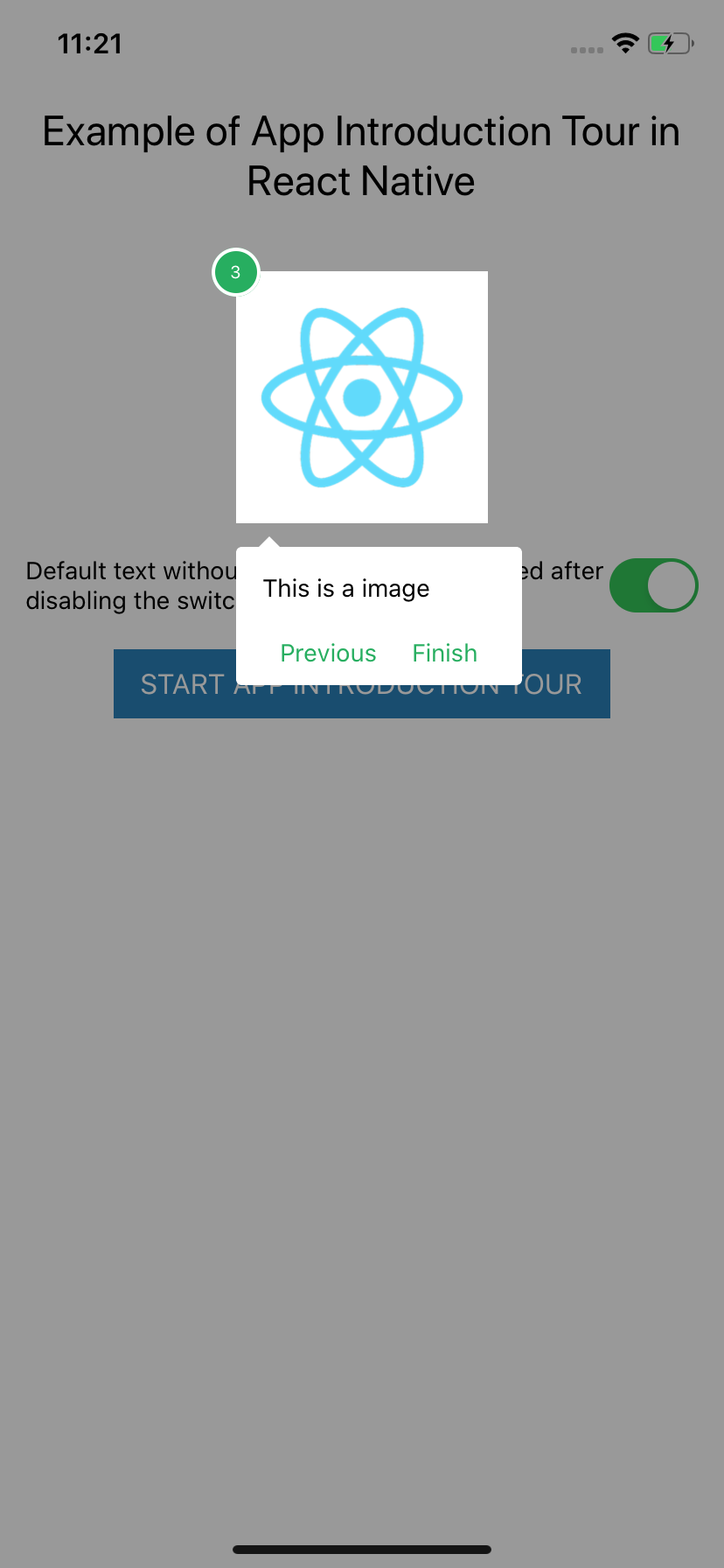
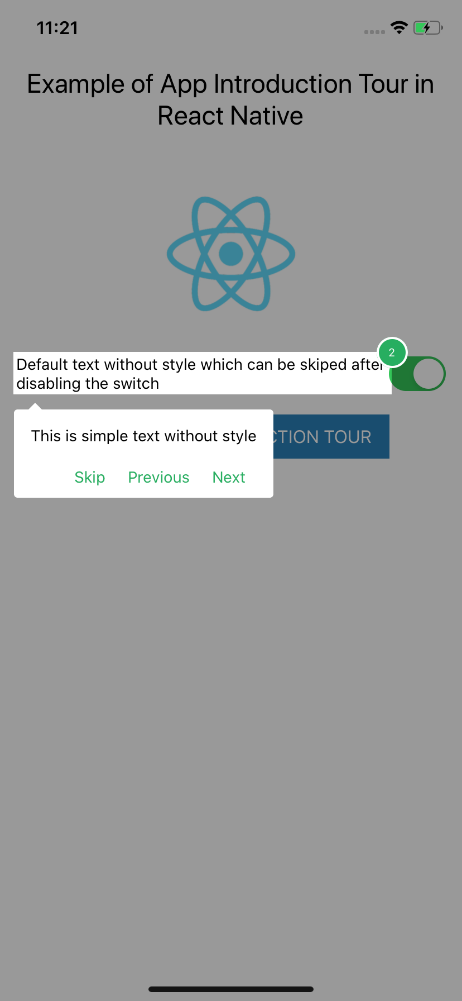
* Khởi động hướng dẫn: Người dùng có thể khởi động hướng dẫn bằng cách nhấn vào một nút "Bắt đầu". Màn hình giới thiệu đầu tiên sẽ được hiển thị, cung cấp thông tin ngắn gọn về nội dung của hướng dẫn.
* Chuyển đổi giữa các bước: Hướng dẫn được chia thành nhiều bước khác nhau. Mỗi bước sẽ có một màn hình riêng với nội dung cụ thể liên quan đến một tính năng hoặc khía cạnh nào đó của ứng dụng.
* Người dùng có thể sử dụng các nút điều hướng như "Tiếp theo", "Quay lại" và "Bỏ qua" để di chuyển giữa các bước. Nút "Bỏ qua" cho phép người dùng rời khỏi hướng dẫn bất cứ lúc nào, điều này rất hữu ích cho những người dùng đã quen thuộc với ứng dụng.
* Kết thúc hướng dẫn: Khi người dùng hoàn thành tất cả các bước, một màn hình kết thúc sẽ được hiển thị, có thể bao gồm các tùy chọn để quay lại trang chính hoặc khởi động lại hướng dẫn nếu cần thiết.

### Lợi ích của Step by step walkthrough

* Cải thiện trải nghiệm người dùng: Hướng dẫn từng bước giúp người dùng mới dễ dàng làm quen với giao diện và tính năng của ứng dụng, giảm sự bối rối khi mới sử dụng ứng dụng.
* Giảm thiểu thời gian làm quen ứng dụng: Walkthrough hướng dẫn người dùng nhanh chóng tìm hiểu cách sử dụng ứng dụng mà không cần phải tự mày mò, giúp tiết kiệm thời gian và công sức trong việc làm quen với ứng dụng.
* Tăng tương tác và giảm tỷ lệ bỏ cuộc.

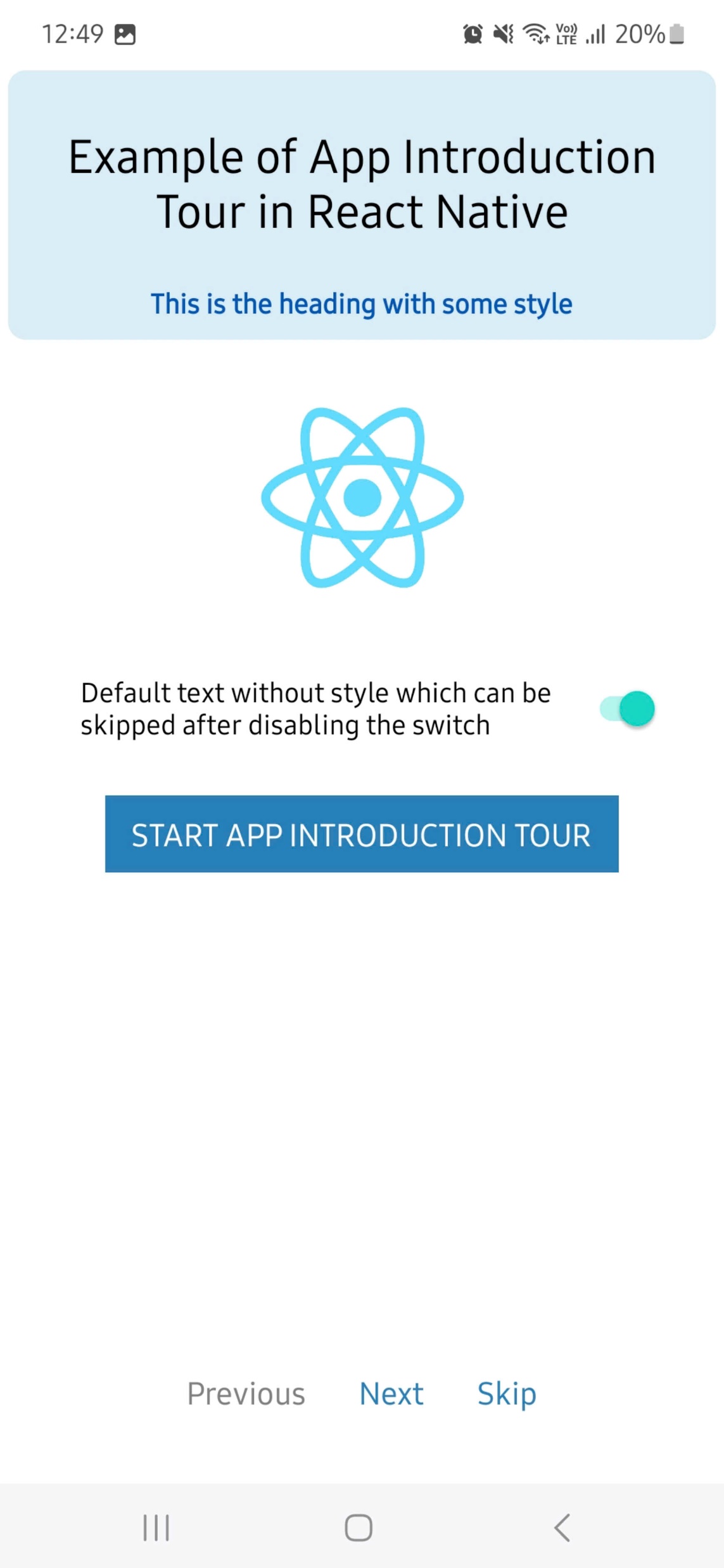
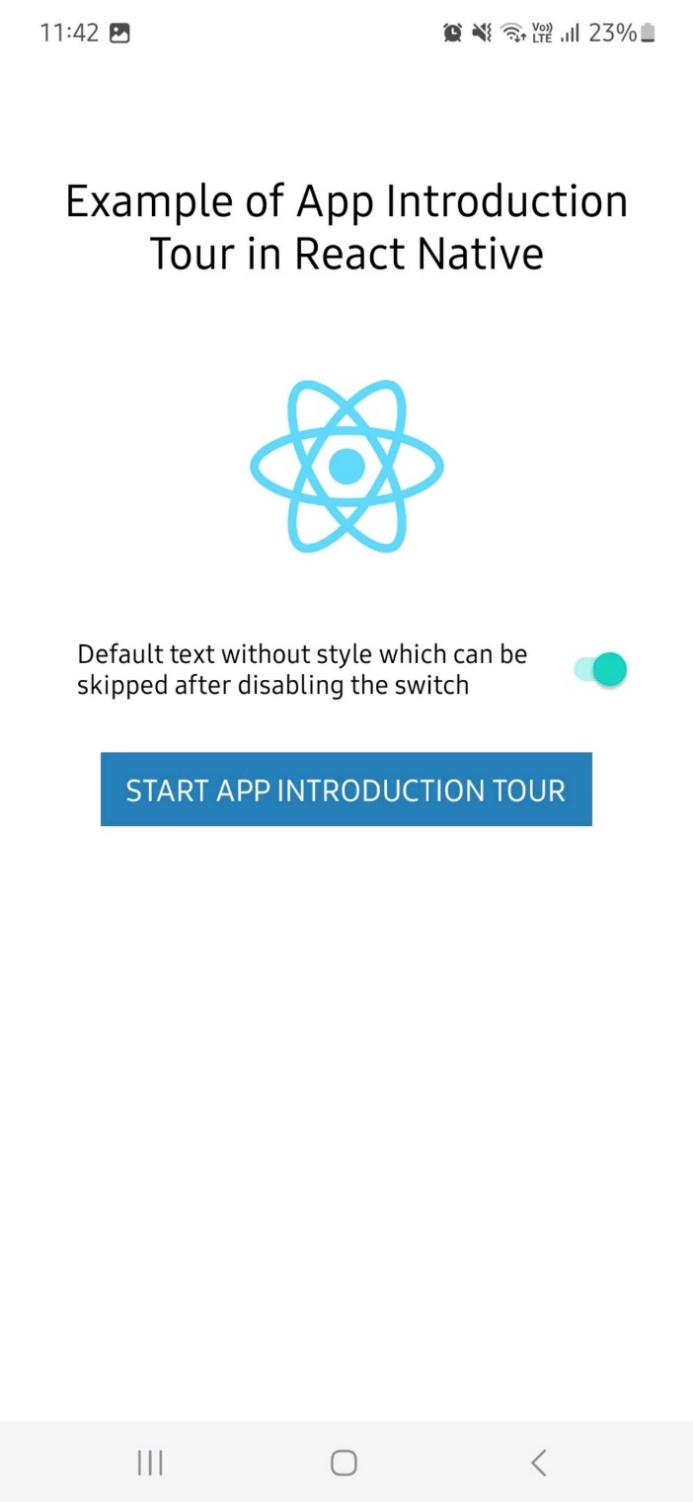
### Hình ảnh demo

* Hình ảnh demo sử dụng thư viện **react-native-copilot**:

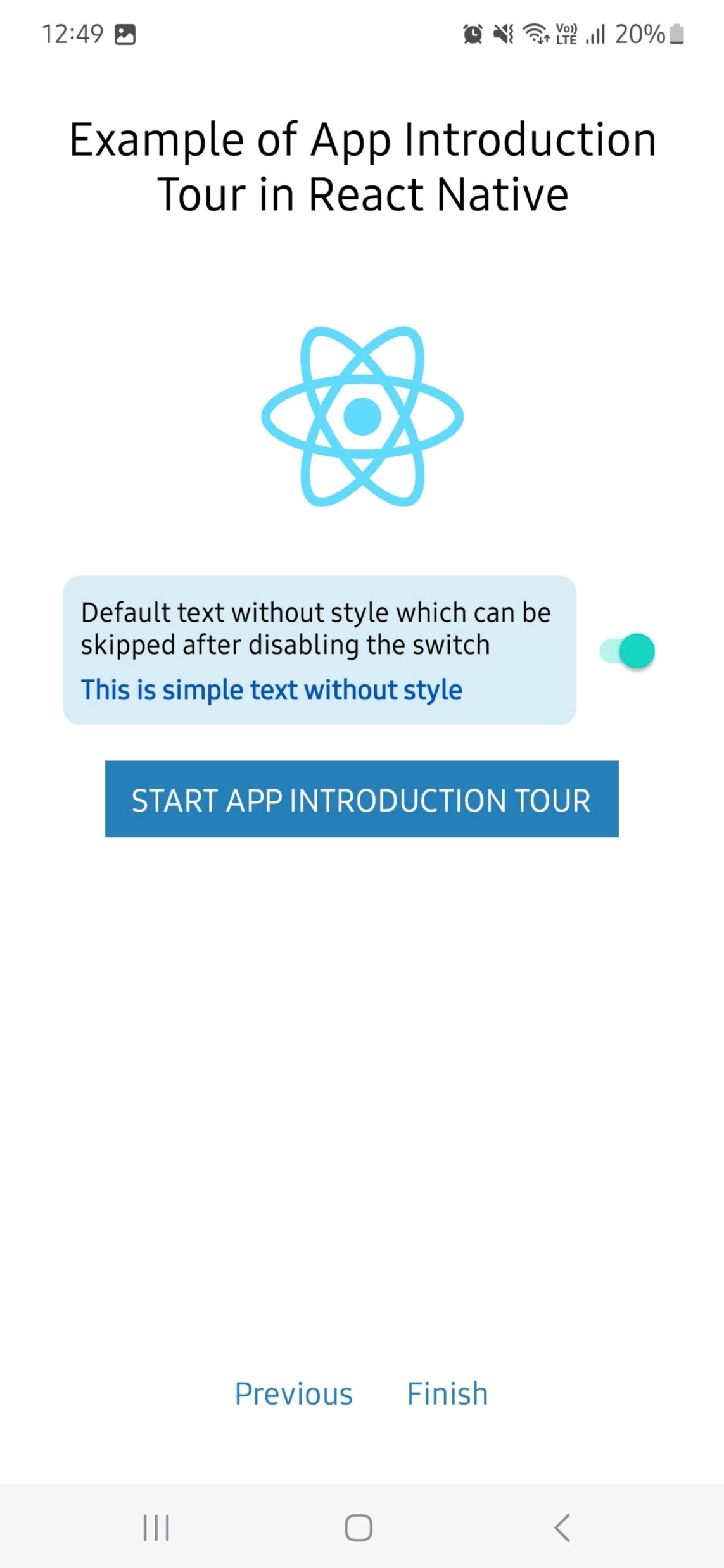
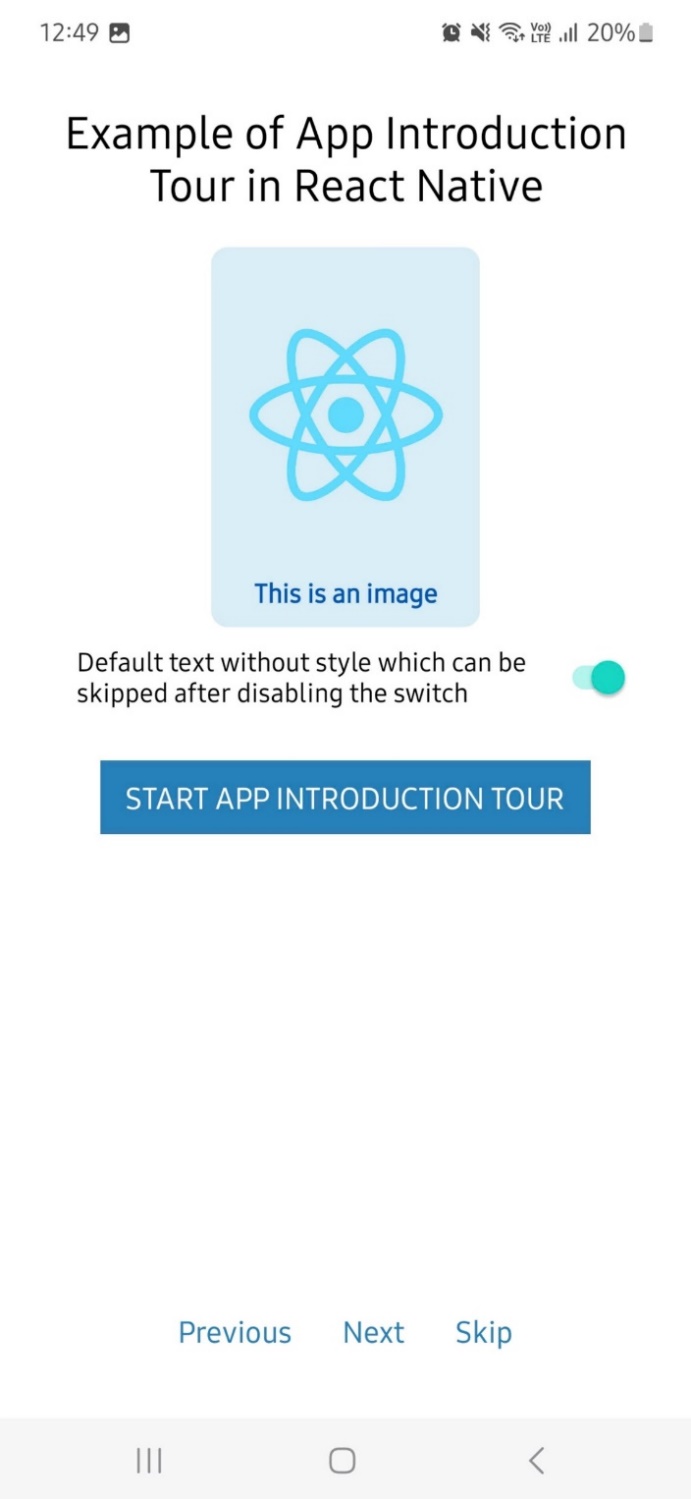


Hình 1.9. Demo Step by Step Walkthrough sử dụng react-native-copilot

* Hình ảnh demo cho phần code thay thế trên Expo Go sử dụng các thành phần React Native sẵn có:



Hình 1.10. Demo Step by Step Walkthrough sử dụng các thành phần React Native sẵn có (1)



Hình 1.11. Demo Step by Step Walkthrough sử dụng các thành phần React Native sẵn có (2)

### Source code

* Link source code gốc: <https://aboutreact.com/step-by-step-app-introduction-in-react-native/>
* Source code được điều chỉnh sử dụng các thành phần React Native sẵn có:

import React, { useState } from 'react';

import {

  SafeAreaView,

  StyleSheet,

  Text,

  Image,

  View,

  TouchableOpacity,

  Switch,

} from 'react-native';

**const** App = () **=>** {

**const** [secondStepActive, setSecondStepActive] = useState(true);

**const** [step, setStep] = useState(0); *// State to track the current step, starting from 0*

  return (

    <SafeAreaView style={styles.container}>

      <View style={styles.innerContainer}>

        {*/\* Step 1: Show the heading \*/*}

        <View style={[styles.titleContainer, step === 1 && styles.highlight]}>

          <Text style={styles.title}>

            Example of App Introduction Tour in React Native

          </Text>

          {step === 1 && <Text style={styles.tooltip}>This is the heading with some style</Text>}

        </View>

        {*/\* Step 2: Show the image \*/*}

        <View style={[styles.imageContainer, step === 2 && styles.highlight]}>

          <Image

            source={{

              uri: 'https://raw.githubusercontent.com/AboutReact/sampleresource/master/react\_logo.png',

            }}

            style={styles.profilePhoto}

          />

          {step === 2 && <Text style={styles.tooltip}>This is an image</Text>}

        </View>

        {*/\* Step 3: Show the normal text with switch \*/*}

        <View style={styles.activeSwitchContainer}>

          <View style={[styles.textContainer, step === 3 && styles.highlight]}>

            <Text>Default text without style which can be skipped after disabling the switch</Text>

            {step === 3 && (

              <Text style={styles.tooltip}>

                This is simple text without style

              </Text>

            )}

          </View>

          <View style={{ flexGrow: 1 }} />

          <Switch

            onValueChange={(value) **=>** setSecondStepActive(value)}

            value={secondStepActive}

          />

        </View>

        {*/\* Button to start the walkthrough \*/*}

        <View style={styles.middleView}>

          <View style={styles.buttonContainer}>

            {*/\* {step === 0 && ( \*/*}

              <TouchableOpacity style={styles.button} onPress={() **=>** setStep(1)}>

                <Text style={styles.buttonText}>START APP INTRODUCTION TOUR</Text>

              </TouchableOpacity>

            {*/\* )} \*/*}

          </View>

        </View>

        {*/\* Control to navigate steps \*/*}

        {step > 0 && (

          <View style={styles.navigationContainer}>

            {step === 3 ? (

*// Last step navigation*

              <>

                <TouchableOpacity

                  onPress={() **=>** setStep((prev) **=>** Math.max(prev - 1, 1))}

                >

                  <Text style={styles.navigationText}>Previous</Text>

                </TouchableOpacity>

                <TouchableOpacity onPress={() **=>** setStep(0)}>

                  <Text style={styles.navigationText}>Finish</Text>

                </TouchableOpacity>

              </>

            ) : (

*// Navigation for steps other than the last one*

              <>

                <TouchableOpacity

                  onPress={() **=>** setStep((prev) **=>** Math.max(prev - 1, 1))}

                  disabled={step === 1}

                >

                  <Text style={[styles.navigationText, step === 1 && styles.disabledText]}>

                    Previous

                  </Text>

                </TouchableOpacity>

                <TouchableOpacity

                  onPress={() **=>** setStep((prev) **=>** Math.min(prev + 1, 4))}

                >

                  <Text style={styles.navigationText}>Next</Text>

                </TouchableOpacity>

                <TouchableOpacity onPress={() **=>** setStep(0)}>

                  <Text style={styles.navigationText}>Skip</Text>

                </TouchableOpacity>

              </>

            )}

          </View>

        )}

      </View>

    </SafeAreaView>

  );

};

**const** styles = StyleSheet.create({

  container: {

    flex: 1,

    backgroundColor: '#fff', *// Keeping the main background white*

    alignItems: 'center',

    paddingTop: 40,

  },

  innerContainer: {

    flex: 1,

    width: '100%',

    alignItems: 'center',

  },

  titleContainer: {

    alignItems: 'center',

  },

  title: {

    fontSize: 24,

    textAlign: 'center',

    margin: 20,

  },

  profilePhoto: {

    width: 140,

    height: 140,

    borderRadius: 70,

    marginVertical: 20,

  },

  middleView: {

    flex: 1,

    alignItems: 'center',

  },

  buttonContainer: {

    alignItems: 'center',

  },

  button: {

    backgroundColor: '#2980b9',

    paddingVertical: 10,

    paddingHorizontal: 15,

  },

  buttonText: {

    color: 'white',

    fontSize: 16,

  },

  activeSwitchContainer: {

    flexDirection: 'row',

    justifyContent: 'space-between',

    marginBottom: 20,

    alignItems: 'center',

    paddingHorizontal: 40,

  },

  highlight: {

    backgroundColor: '#d9edf7', *// Highlight color without border*

    borderRadius: 10, *// Added border radius*

    padding: 10,

  },

  tooltip: {

    marginTop: 5,

    color: '#0056b3', *//2980b9*

    fontWeight: 'bold',

  },

  textContainer: {

    padding: 10,

  },

  imageContainer: {

    alignItems: 'center',

  },

  navigationContainer: {

    flexDirection: 'row',

    justifyContent: 'center', *// Centering the buttons*

    width: '100%', *// Full width to help with centering*

    marginTop: 20,

    marginBottom: 40, *// Added margin to create space at the bottom*

  },

  navigationText: {

    color: '#2980b9', *// Text color for navigation options*

    fontSize: 16,

    marginHorizontal: 15, *// Adjusted margin for better spacing*

  },

  disabledText: {

    color: 'grey', *// Color for disabled Previous button*

  },

});

export default App;

# Tìm hiểu về Splash Screens

## Định nghĩa

Splash Screen (hay màn hình khởi động) là màn hình đầu tiên mà người dùng nhìn thấy khi mở một ứng dụng trên thiết bị di động. Màn hình này thường hiển thị logo, tên ứng dụng hoặc một hình ảnh liên quan đến ứng dụng. Mục đích chính là chào mừng người dùng và tạo ấn tượng ban đầu khi ứng dụng đang tải dữ liệu hoặc khởi chạy.

## Mục đích:

Giới thiệu ứng dụng: Hiển thị logo và tên ứng dụng giúp người dùng nhận diện ứng dụng dễ dàng hơn.

Tạo ấn tượng: Một màn hình khởi động đẹp mắt sẽ giúp người dùng có cảm giác chuyên nghiệp và tạo cảm giác thích thú khi sử dụng

Giảm thời gian chờ: Splash Screen giúp giảm cảm giác chờ của người dùng khi tải ứng dụng hoặc xử lý các tác vụ nền.

## Thành phần của Splash Screen:

Logo ứng dụng: Là yếu tố nổi bật trên màn hình, giúp nhận diện ứng dụng

Màu Nền: là màu đặc trưng của ứng dụng

Thông Điệp Ngắn (Tùy Chọn): Có thể là khẩu hiệu, tên ứng dụng hoặc một câu chào ngắn gọn.

## Phân loại Splash Screen: gồm hai loại Splash Screen phổ biến:

Static Splash Screen: Màn hình tĩnh chỉ hiển thị hình ảnh hoặc logo đơn giản trong một khoảng thời gian ngắn.

Animated Splash Screen: Màn hình có các hiệu ứng chuyển động (ví dụ: logo xuất hiện từ từ, chuyển động, hoặc hiệu ứng fade-in/fade-out).

## Animated Splash Screen trong React Native

Trong React Native, **Animated Splash Screen** có thể kết hợp các hiệu ứng chuyển động để giúp ứng dụng thêm sinh động, tạo cảm giác mượt mà khi người dùng chờ đợi ứng dụng tải nội dung.

Link tài liệu: <https://aboutreact.com/animated-splash-screen/>

Hiển thị màn hình

**A screen shot of a cell phone

Description automatically generated**

Hình 2.1. Màn hình hiển thị hiệu ứng Animated Splash Screen

## Animation Splash Screen with Zoom Effect

Animation Splash Screen with Zoom Effect là một kiểu Splash Screen có hiệu ứng phóng to, trong đó logo hoặc hình ảnh chính của ứng dụng sẽ dần dần được phóng to ngay khi ứng dụng bắt đầu.

Link tài liệu: [<https://aboutreact.com/animation-splash-screen-with-zoom-effect/>](https://aboutreact.com/animation-splash-screen-with-zoom-effect/)

Hiển thị màn hình:

A blue screen with white text

Description automatically generated

Hình 2.2. Màn hình hiển thị hiệu ứng Animated Splash Screen with Zoom Effect

## Lợi ích:

Tăng cường trải nghiệm người dùng: Một màn hình khởi động hấp dẫn giúp tạo thiện cảm và giữ chân người dùng ở lại ứng dụng lâu hơn.

Xây dựng hình ảnh ứng dụng: Thông qua logo và hình ảnh ứng dụng, Splash Screen giúp tăng cường khả năng nhận diện và ghi nhớ ứng dụng.

Che giấu quá trình tải: Một số ứng dụng có thời gian tải ban đầu lâu (do tải dữ liệu hoặc cài đặt kết nối), Splash Screen giúp che giấu thời gian này một cách tinh tế.

# Tìm Hiểu Về Blur & Gradient Components

Trong thiết kế ứng dụng, **Blur (hiệu ứng mờ)** và **Gradient (hiệu ứng chuyển màu)** là những thành phần phổ biến giúp tạo chiều sâu, làm nổi bật các yếu tố giao diện, và mang lại cảm giác tinh tế, hiện đại.

## Blur Component

Để tạo hiệu ứng mờ (Blur Effect), có thể sử dụng thư viện @react-native-community/blur, giúp tạo ra các lớp nền mờ phía sau các thành phần.

Cài đặt thư viện bằng lệnh sau:

*npm install @react-native-community/blur*

Tham khảo code mẫu tại link: <https://aboutreact.com/blur-background/>

Màn hình ứng dụng **trước** khi áp dụng hiệu ứng làm mờ (Blur Effect)

A group of white flowers with yellow centers

Description automatically generated

Hình 3.1. Màn hình ứng dụng trước khi áp dụng hiệu ứng làm mờ

Màn hình ứng dụng **sau** khi áp dụng hiệu ứng làm mờ (Blur Effect)

**- Xlight:** tạo hiệu ứng mờ đục và sáng hơn so với light. Bằng cách đảo ngược biến *tintColor,* dải màu ['#ffffff', '#000000'] sẽ đảo ngược thành ['#000000', '#ffffff']

**- Light:** Chế độ này tintColor không bị dảo ngược dải màu nên tạo cảm giác làm mờ trong hơn so với xlight.

**- Dark:** Chế độ này giống như Light, tuy nhiên độ sáng lúc này đã được hạ thấp hơn nhiều, phù hợp để sử dụng trong môi trường ánh sáng yếu

|  |  |  |
| --- | --- | --- |
|  |  |  |
| Hình 3.2. Xlight | Hình 3.3. Light | Hình 3.4. Dark |

**Ý nghĩa:**

**- Tạo Hiệu Ứng Nổi Bật:** Làm mờ nền giúp người dùng tập trung vào các thành phần trên màn hình, như tiêu đề hoặc thông báo.

**- Cải Thiện Trải Nghiệm Người Dùng:** Mang lại giao diện hiện đại, mượt mà, dễ nhìn trong các thiết kế ứng dụng phức tạp.

**- Tăng Chiều Sâu Cho Giao Diện:** Tạo cảm giác phân lớp và chiều sâu cho giao diện, giúp ứng dụng trở nên chuyên nghiệp hơn.

## Linear Gradient Component

Linear Gradient là một hiệu ứng phổ biến được sử dụng để chuyển đổi màu sắc mượt mà hơn, tạo sự hài hòa, nổi bật trong giao diện của ứng dụng. Bằng cách sử dụng Linear Gradient Component trong React Native, chúng ta có thể tạo ra những màn hình có nền gradient bắt mắt hoặc tạo hiệu ứng màu sắc cho các thành phần UI khác nhau.

Cài đặt thư viện linear gradient bằng lệnh sau:

*npm install react-native-linear-gradient*

Tham khảo code mẫu tại link: <https://aboutreact.com/react-native-linear-gradient/>

**Bao gồm:**

**- Simple Linear Gradient Background:** Chuyển đổi hiệu ứng màu Gradient đơn giản nhất. Các màu sắc được ngăn cách bởi một đường thẳng theo phương ngang và có chiều dài xuyên suốt background ứng dụng.

A computer screen with text and numbers

Description automatically generated

Hình 3.5. Code Simple Gradient

**- Horizontal Gradient Background:** là dạng gradient có màu sắc chuyển tiếp theo chiều ngang, từ trái sang phải (hoặc ngược lại). Với **start={{ x: 0, y: 0 }} và end={{ x: 1, y: 0 }}**: Các điểm bắt đầu và kết thúc của gradient được thiết lập theo chiều ngang, với x: 0 là điểm bên trái và x: 1 là điểm bên phải.

A computer screen with text and numbers

Description automatically generated

Hình 3.6. Code Horizontal Gradient

**- Location Gradient Background:** là dạng gradient với các màu chuyển tiếp được xác định bởi các điểm được chỉ định trước. Với việc đặt các "điểm dừng màu" (color stops) tại các vị trí cụ thể, có thể tạo hiệu ứng gradient với những biến đổi sắc màu độc đáo, tạo điểm nhấn rõ ràng trong các vùng nhất định của UI ứng dụng.

A computer screen shot of text

Description automatically generated

Hình 3.7. Code Location Gradient

Với **locations={[0, 0. 5, 0.8]}**: Đặt các điểm màu tại 0%, 50%, và 80% chiều dài gradient, giúp tạo hiệu ứng màu sắc độc đáo và nổi bật.

**Kết quả hiển thị trên màn hình:**

A group of colorful rectangular objects

Description automatically generated with medium confidence

Hình 3.8. Màn hình hiển thị hiệu ứng Linear Gradient

**Ý nghĩa**

**- Tạo điểm nhấn:** Linear Gradient Component có thể tạo ra các điểm nhấn trong giao diện với các màu chuyển tiếp khác nhau, giúp thu hút sự chú ý của người dùng

**- Tăng tính thẩm mỹ:** Hiệu ứng gradient mượt mà tạo cảm giác dễ chịu, hài hòa và giúp giao diện trở nên tinh tế hơn.

**- Tuỳ chỉnh dễ dàng:** Chỉ cần thay đổi một vài giá trị màu sắc, có thể dễ dàng tạo ra nhiều kiểu nền gradient khác nhau, phù hợp với phong cách của ứng dụng.

# KẾT LUẬN

Thông qua bài này, nhóm đã tìm hiểu được mục đích, ý nghĩa và cách ứng dụng các thành phần như App Intro Slider, Màn hình Splash Screen và hiệu ứng Blur & Gradient trong React Navive, với mục đích nâng cao trải nghiệm người dùng, tạo ấn tượng chuyên nghiệp và thu hút người dùng ngay từ ban đầu.

Ngoài ra, việc ứng dụng hiệu quả App Intro Slider, Splash Screen và các hiệu ứng Blur & Gradient, còn giúp xây dựng được một nền tảng giao diện đẹp mắt và đáp ứng được nhu cầu trải nghiệm mượt mà của người dùng.

Qua đó, nhóm nhận thấy tầm quan trọng của việc đầu tư vào giao diện khởi động và hiệu ứng thị giác, giúp ứng dụng nổi bật và thân thiện hơn với người dùng. Trong tương lai, nhóm sẽ tiếp tục khám phá thêm các kỹ thuật và thành phần mới nhằm nâng cao hơn nữa trải nghiệm người dùng trong React Native.

Tuy nhiên do giới hạn về mặt thời gian, nhóm chưa nghiên cứu sâu hơn các tính năng nâng cao nhằm tối ưu trải nghiệm người dùng, đặc biệt là việc triển khai các hiệu ứng chuyển động cải thiện tính thẩm mỹ của ứng dụng.

# TÀI LIỆU THAM KHẢO

|  |  |
| --- | --- |
| [1] | https://aboutreact.com/react-native-view-pager/ [Online]. [Accessed 02 11 2024]. |
| [2] | https://aboutreact.com/react-native-app-intro-slide/ [Online]. [Accessed 02 11 2024]. |
| [3] | https://aboutreact.com/step-by-step-app-introduction-in-react-native/ [Online]. [Accessed 02 11 2024]. |
| [4] | https://aboutreact.com/animated-splash-screen/ [Online]. [Accessed 02 11 2024]. |
| [5] | https://aboutreact.com/animation-splash-screen-with-zoom-effect/ [Online]. [Accessed 02 11 2024]. |
| [6] | https://aboutreact.com/blur-background/ [Online]. [Accessed 02 11 2024]. |
| [7] | https://aboutreact.com/react-native-linear-gradient/ [Online]. [Accessed 02 11 2024]. |