**Image Search App Documentation**

**Overview**

This React Native app allows users to search for images using the SerpApi service, display the images, and view additional details in a modal when an image is clicked. The app also includes a category selection feature to filter images based on different categories.

**Prerequisites**

1. **Node.js and npm**: Ensure you have Node.js and npm installed. You can download them from [nodejs.org](https://nodejs.org/).
2. **React Native CLI**: Install the React Native CLI globally if you haven't already:

npm install -g react-native-cli

1. **Android Studio or Xcode**: Install Android Studio for Android development or Xcode for iOS development. Follow the official React Native documentation for setting up the development environment.

**Setting Up the Project**

1. **Create a New React Native Project**

Run the following command to initialize a new React Native project:

npx react-native init ImageSearchApp

1. **Navigate to the Project Directory**

cd ImageSearchApp

1. **Install Required Dependencies** Install the required libraries, including Axios for HTTP requests and @react-native-picker/picker for the dropdown menu:

npm install axios @react-native-picker/picker

**Project Structure**

* App.jsx: Main application component.
* serpApi.js: Contains the function to fetch images from SerpApi.
* ImageModal.jsx: Displays the selected image in a modal with additional details.

**Code Explanation**

**1. serpApi.js**

This file contains the function to fetch images from the SerpApi service.

javascript

import axios from 'axios';

const API\_KEY = 'YOUR\_SERPAPI\_KEY'; // Replace with your SerpApi key

const BASE\_URL = 'https://serpapi.com/search';

export const fetchImages = async (query, category) => {

try {

const params = {

engine: 'google',

q: `${query} ${category !== 'All' ? category : ''}`,

tbm: 'isch',

api\_key: API\_KEY,

};

const response = await axios.get(BASE\_URL, { params });

return response.data.images\_results;

} catch (error) {

console.error('Error fetching images:', error);

throw error;

}

};

**2. ImageModal.jsx**

This component displays the selected image in a modal with its details.

javascript

import React from 'react';

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

const ImageModal = ({ image, onClose }) => {

if (!image) return null;

return (

<Modal

transparent={true}

animationType="slide"

visible={!!image}

onRequestClose={onClose}

>

<View style={styles.modalContainer}>

<TouchableOpacity style={styles.closeButton} onPress={onClose}>

<Text style={styles.closeButtonText}>Close</Text>

</TouchableOpacity>

<Image

source={{ uri: image.thumbnail }}

style={styles.image}

resizeMode="contain"

/>

<Text style={styles.imageDetails}>Height: {image.height}</Text>

<Text style={styles.imageDetails}>Width: {image.width}</Text>

</View>

</Modal>

);

};

const styles = StyleSheet.create({

modalContainer: {

flex: 1,

justifyContent: 'center',

alignItems: 'center',

backgroundColor: 'rgba(0, 0, 0, 0.8)',

},

closeButton: {

position: 'absolute',

top: 40,

right: 20,

padding: 10,

backgroundColor: '#fff',

borderRadius: 5,

},

closeButtonText: {

fontSize: 16,

fontWeight: 'bold',

},

image: {

width: '90%',

height: '70%',

},

imageDetails: {

color: '#fff',

marginTop: 10,

fontSize: 16,

},

});

export default ImageModal;

**3. App.jsx**

The main component of the app, handling the UI and interaction.

javascript

import React, { useState } from 'react';

import {

View,

TextInput,

Button,

FlatList,

Image,

TouchableOpacity,

StyleSheet,

} from 'react-native';

import { Picker } from '@react-native-picker/picker';

import { fetchImages } from './serpApi';

import ImageModal from './ImageModal';

const categories = ['All', 'Nature', 'Technology', 'People', 'Animals', 'Travel'];

const App = () => {

const [query, setQuery] = useState('');

const [images, setImages] = useState([]);

const [selectedImage, setSelectedImage] = useState(null);

const [category, setCategory] = useState('All');

const handleSearch = async () => {

try {

const result = await fetchImages(query, category);

setImages(result);

} catch (error) {

console.error('Error fetching images:', error);

}

};

const renderImage = ({ item }) => (

<TouchableOpacity onPress={() => setSelectedImage(item)}>

<Image source={{ uri: item.thumbnail }} style={styles.image} />

</TouchableOpacity>

);

return (

<View style={styles.container}>

<TextInput

style={styles.input}

placeholder="Search for images..."

value={query}

onChangeText={setQuery}

/>

<Picker

selectedValue={category}

style={styles.picker}

onValueChange={(itemValue) => setCategory(itemValue)}

>

{categories.map((cat) => (

<Picker.Item key={cat} label={cat} value={cat} />

))}

</Picker>

<Button title="Search" onPress={handleSearch} />

<FlatList

data={images}

renderItem={renderImage}

keyExtractor={(item) => item.link}

numColumns={2}

contentContainerStyle={styles.imageList}

/>

{selectedImage && (

<ImageModal

image={selectedImage}

onClose={() => setSelectedImage(null)}

/>

)}

</View>

);

};

const styles = StyleSheet.create({

container: {

flex: 1,

padding: 16,

},

input: {

height: 40,

borderColor: 'gray',

borderWidth: 1,

marginBottom: 16,

paddingHorizontal: 8,

},

picker: {

height: 50,

width: '100%',

marginBottom: 16,

},

image: {

width: 150,

height: 150,

margin: 8,

},

imageList: {

alignItems: 'center',

},

});

export default App;

**Running the Project**

1. **Start the Development Server**

npx react-native start

1. **Run on Android Emulator/Device**

npx react-native run-android

1. **Run on iOS Simulator/Device**

npx react-native run-ios

**Testing the Application**

1. **Open the App**: You should see a text input, a category picker, a search button, and a list of images.
2. **Search for Images**: Enter a search query and select a category, then click the search button to fetch images.
3. **View Image Details**: Click on any image to view it in a modal with its height and width details. Click the close button to dismiss the modal.

**Common Issues and Troubleshooting**

* **401 Unauthorized Error**: Ensure that you have correctly replaced 'YOUR\_SERPAPI\_KEY' in serpApi.js with your actual SerpApi key.
* **Picker Not Found**: Make sure you have installed @react-native-picker/picker and imported it correctly in App.jsx.
* **Images Not Displaying**: Verify the structure of the image object and ensure that the thumbnail, height, and width properties are being passed correctly.

**Conclusion**

This documentation provides a comprehensive guide to setting up and running the image search app. If you encounter any issues or need further assistance, refer to the React Native documentation or the SerpApi documentation.