Vitor Amadeu Souza

Introdução ao

React Native

No ambiente Expo Snack

Parte XXII

© 2024 by Cerne Tecnologia e Treinamento Ltda.

© 2024 by Vitor Amadeu Souza

Nenhuma parte desta publicação poderá ser reproduzida sem autorização prévia e escrita de **Cerne Tecnologia e Treinamento Ltda.** Este livro publica nomes comerciais e marcas registradas de produtos pertencentes a diversas companhias. O editor utiliza as marcas somente para fins editoriais e em benefício dos proprietários das marcas, sem nenhuma intenção de atingir seus direitos.

Outubro de 2024

Direitos reservados por:
Cerne Tecnologia e Treinamento Ltda
Produção: Cerne Tecnologia e Treinamento
E-mail da Empresa: cerne@cerne-tec.com.br
Home Page: www.cerne-tec.com.br.com.br
Atendimento ao Consumidor: sac@cerne-tec.com.br
Contato com o Autor: vitor@cerne-tec.com.br

FEITO NO BRASIL

“Venha também sobre mim a tua benignidade, ó Senhor, e a tua salvação, segundo a tua palavra.”

Sl 119:41

Cerne Tecnologia

A Cerne Tecnologia tem uma equipe preparada para desenvolvimento de projetos eletrônicos em diversas áreas: Médica, Entretenimento, Industrial, Robótica, Científica, Automobilística, Aeronáutica, etc. Trabalhamos com tecnologia microcontrolada usando o PIC, ARM, AVR, 8051, dsPIC, PIC24, PIC32 além do Arduino, Raspberry, Beaglebone etc. Desenvolvemos o projeto desde sua concepção até a entrega do produto final, passando pelas etapas de esquema elétrico, protótipo e desenvolvimento de circuito impresso.

Desenvolvemos aplicativos para smartphones/tablets Android, iOS, Blackberry, Windows Phone e no desenvolvimento de softwares a nível PC para plataforma Windows, usando ferramentas como o Visual Basic, C# e C++.

Atuamos na parte de montagem de placas, onde podemos fornecer ambos os serviços de desenvolvimento de projetos e produção ou apenas um destes.

Desenvolvemos esquemas elétricos e layout de PCI, tanto em tecnologia convencional como SMD.

Temos a flexibilidade de customizar um de nossos produtos, de modo a atender a uma necessidade específica do cliente, tornando o custo de desenvolvimento menor se comparado a construção de um projeto desde a sua fase inicial.

Desenvolvemos e fornecemos kits didáticos para diversos microcontroladores além de apostilas, livros e e-books.

Na hora de desenvolver um projeto ou equipar seu laboratório não hesite em nos contatar. Entre em contato conosco através do endereço cerne-tec.com.br para obter mais informações.

Cerne
Conhecimento para o Desenvolvimento

cerne-tec.com.br

Sumário

Capítulo I – Metodologia de desenvolvimento.....6

1. Introdução.....6

Capítulo II – Programação em React Native.....7

1. Acesso por biometria 7

2. Tirando foto e enviando por e-mail.....14

3. Filtro e busca.....16

Capítulo I
Metodologia de desenvolvimento

1. Introdução

Neste livro, você encontrará uma parte da série dedicada ao desenvolvimento de projetos com React Native. Ao longo desta obra, diversos experimentos serão apresentados conforme o sumário, permitindo um aperfeiçoamento contínuo no uso deste framework para a construção de interfaces responsivas com foco em dispositivos móveis.

Capítulo II

Programação em React Native

1. Acesso por biometria

O código apresentado é um componente funcional chamado BiometricLoginScreen, desenvolvido com React Native, que implementa um sistema de autenticação biométrica utilizando a biblioteca expo-local-authentication.

No início, o componente importa os módulos necessários, como React, useState para gerenciar o estado, e componentes básicos do React Native, como View, Text, Button e StyleSheet. O estado authenticated é definido com o useState, inicialmente como null, para rastrear o resultado da autenticação.

A função authenticateUser é responsável por gerenciar o processo de autenticação. Primeiro, ela verifica se o dispositivo tem hardware de biometria disponível usando LocalAuthentication.hasHardwareAsync(). Se não houver hardware, o estado authenticated é atualizado para false. Em seguida, a função verifica se há métodos biométricos cadastrados no dispositivo com

LocalAuthentication.isEnrolledAsync(). Se não houver nenhum método cadastrado, o estado também é atualizado para false.

Caso o hardware esteja disponível e métodos biométricos estejam cadastrados, a função chama LocalAuthentication.authenticateAsync(), que exibe uma mensagem solicitando ao usuário que use sua biometria para autenticar. O resultado da autenticação é então armazenado no estado authenticated.

Na renderização do componente, um View é utilizado para exibir mensagens de status de autenticação ao usuário. Se authenticated for null, a mensagem "Autenticando..." é exibida. Se o usuário for autenticado com sucesso, a mensagem "Acesso autorizado" é mostrada, enquanto "Acesso negado" é exibida se a autenticação falhar. O fundo do View muda para verde se a autenticação for bem-sucedida e para vermelho se falhar. Por fim, um botão "Tentar Novamente" é fornecido para que o usuário possa repetir o processo de autenticação.

Os estilos são definidos utilizando StyleSheet.create(), que aplica formatação ao layout, texto e cores de fundo, garantindo uma apresentação visual adequada das mensagens de autenticação.

```
import React, { useState } from 'react';
import { View, Text, Button, StyleSheet } from 'react-native';
import * as LocalAuthentication from 'expo-local-authentication';


const BiometricLoginScreen = () => {
  const [authenticated, setAuthenticated] = useState(null);


  const authenticateUser = async () => {
    const hasHardware = await LocalAuthentication.hasHardwareAsync();
    if (!hasHardware) {
      setAuthenticated(false);
      return;
    }


    const isEnrolled = await LocalAuthentication.isEnrolledAsync();
    if (!isEnrolled) {
      setAuthenticated(false);
      return;
    }


    const result = await LocalAuthentication.authenticateAsync({
      promptMessage: 'Use sua biometria para entrar',
    });


    setAuthenticated(result.success);
  };


  return (
    <View style={[styles.container, authenticated === null ? null :
(authenticated ? styles.success : styles.error)]}>
      <Text style={styles.text}>
        {authenticated === null
          ? 'Autenticando...'
          : authenticated
          ? 'Acesso autorizado '
          : 'Acesso negado '}
      </Text>
      <Button title="Tentar Novamente" onPress={authenticateUser} />
    </View>
  );
};


const styles = StyleSheet.create({
  container: {
    flex: 1,
    justifyContent: 'center',
    alignItems: 'center',
```

```
  },
  success: {
    backgroundColor: 'green',
  },
  error: {
    backgroundColor: 'red',
  },
  text: {
    fontSize: 18,
    color: '#fff',
    marginBottom: 20,
  },
});

export default BiometricLoginScreen;
```