

Vitor Amadeu Souza

Introdução ao

React Native

No ambiente Expo Snack

Parte XXVII

© 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.

Novembro 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 PCB, 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-tec.com.br

Sumário

Capítulo I – Metodologia de desenvolvimento.....	6
1. Introdução.....	6
Capítulo II – Programação em React Native.....	7
1. Bloco de notas.....	7
2. Conversor de unidades.....	13
3. Catálogo de Livros.....	18
4. Queima de calorias.....	23

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. Bloco de notas

Este exemplo é um bloco de notas interativo com funcionalidades para adicionar, visualizar e gerenciar notas. Ele permite que o usuário insira texto, adicione uma tag para categorização e inclua uma imagem associada à nota, utilizando a biblioteca expo-image-picker para selecionar imagens da galeria do dispositivo.

A interface é projetada com uma barra de progresso que simula a sincronização das notas entre dispositivos, representada pelo componente `ProgressBarAndroid`. Quando o usuário deseja sincronizar as notas, a barra de progresso é atualizada em incrementos, e uma mensagem de sucesso é exibida ao final do processo.

A estrutura das notas inclui informações como texto, tag e imagem. Cada nota é exibida em uma lista usando `FlatList`, com imagens mostradas em miniaturas. Para melhorar a experiência do

usuário, ao clicar no botão "Escolher Imagem", o aplicativo abre a galeria, e a imagem selecionada é exibida na tela antes de ser associada à nova nota.

Os estilos aplicados garantem uma interface visualmente agradável e organizada, com botões estilizados e campos de entrada intuitivos. O layout do aplicativo é responsivo e oferece uma experiência de usuário suave. O `useState` é usado para gerenciar estados como a lista de notas, o texto da nota, a tag, e o progresso de sincronização.

Essa abordagem cria um aplicativo funcional, oferecendo funcionalidades básicas de um gerenciador de notas com suporte a imagens, categorização e um elemento visual de progresso para simulação de sincronização.

```
import React, { useState } from 'react';
import { View, Text, TextInput, Button, StyleSheet, FlatList,
TouchableOpacity, Image, Alert, ProgressBarAndroid } from 'react-native';
import * as ImagePicker from 'expo-image-picker';
import * as FileSystem from 'expo-file-system';
import { Ionicons } from '@expo/vector-icons';

const App = () => {
  const [notes, setNotes] = useState([]);
  const [noteText, setNoteText] = useState('');
  const [noteTag, setNoteTag] = useState('');
  const [imageUri, setImageUri] = useState(null);
  const [progress, setProgress] = useState(0);
```

```

// Função para adicionar nota
const addNote = () => {
  if (noteText.trim() === '') {
    Alert.alert('Erro', 'Por favor, insira o texto da nota.');
```

return;

```
  }

  const newNote = {
    id: Date.now().toString(),
    text: noteText,
    tag: noteTag,
    image: imageUri,
  };
  setNotes([newNote, ...notes]);
  setNoteText('');
  setNoteTag('');
  setImageUri(null);
};

// Função para escolher imagem
const pickImage = async () => {
  let result = await ImagePicker.launchImageLibraryAsync({
    mediaTypes: ImagePicker.MediaTypeOptions.Images,
    allowsEditing: true,
    quality: 1,
  });

  if (!result.canceled) {
    setImageUri(result.assets[0].uri);
  }
};

// Função para simular a sincronização entre dispositivos
const syncNotes = () => {
  let syncProgress = 0;
  const interval = setInterval(() => {
    if (syncProgress < 100) {
      syncProgress += 10;
      setProgress(syncProgress);
    } else {
      clearInterval(interval);
      Alert.alert('Sincronização Completa', 'Suas notas foram sincronizadas
com sucesso!');
```

}

```
    }, 500);
  };

  return (
    <View style={styles.container}>
```

```

<Text style={styles.title}>Bloco de Notas</Text>

<TextInput
  style={styles.input}
  placeholder="Escreva sua nota"
  value={noteText}
  onChangeText={setNoteText}
  multiline
/>
<TextInput
  style={styles.input}
  placeholder="Tag da nota"
  value={noteTag}
  onChangeText={setNoteTag}
/>

<TouchableOpacity onPress={pickImage} style={styles.button}>
  <Text style={styles.buttonText}>Escolher Imagem</Text>
</TouchableOpacity>

{imageUri && (
  <Image source={{ uri: imageUri }} style={styles.image} />
)}

<Button title="Adicionar Nota" onPress={addNote} />

{notes.length > 0 && (
  <>
    {/*      <Button title="Sincronizar Notas" onPress={syncNotes} />*/}
    <ProgressBarAndroid style={styles.progressBar}
indeterminate={false} progress={progress / 100} />
    </>
  )}

<FlatList
  data={notes}
  keyExtractor={({ item }) => item.id}
  renderItem={({ item }) => (
    <View style={styles.noteContainer}>
      <Text style={styles.noteText}>{item.text}</Text>
      <Text style={styles.noteTag}>Tag: {item.tag}</Text>
      {item.image && <Image source={{ uri: item.image }}
style={styles.noteImage} />}
    </View>
  )}
/>
</View>
);
};

```