Colocar as imagens do App na tela retirada

const bgImage = require("../assets/background2.png");

const appIcon=require("../assets/appIcon.png");

const appName=require("../assets/appName.png");

colocar os atibutos abaixo na construtora.

```
constructor(props) {
  super(props);
  this.state = {
   domState: "normal",
   hasCameraPermissions: null,
   scanned: false,
   scannedData: "",
   bookId:"",
   studentId:"",
   bookName:"",
   studentName:""
  };
 }
```

Modificar o método handleBarCodeScanned

```
handleBarCodeScanned = async ({ type, data }) => {
  const { domState } = this.state;

  if (domState === "bookId") {
   this.setState({
    bookId: data,
    domState: "normal",
    scanned: true
   });
  } else if (domState === "studentId") {
   this.setState({
    studentId: data,
    domState: "normal",
    scanned: true
   });
  }
 };
```

Modificar o metodo handleTransaction

```
handleTransaction = async () => {
 var { bookId, studentId } = this.state;
 await this.getBookDetails(bookId);
 await this.getStudentDetails(studentId);

 db.collection("books")
  .doc(bookId)
  .get()
  .then(doc => {
   var book = doc.data();
   if (book.is_book_available) {
    var { bookName, studentName } = this.state;
    this.initiateBookIssue(bookId, studentId, bookName, studentName);

    // Apenas para usuários do Android
    ToastAndroid.show("Livro entregue para o aluno!", ToastAndroid.SHORT);

    // Alert.alert("Book issued to the student!");
   } else {
    var { bookName, studentName } = this.state;
    this.initiateBookReturn(bookId, studentId, bookName, studentName);

    // Apenas para usuários do Android
    ToastAndroid.show("Livro retornado à biblioteca!", ToastAndroid.SHORT);

    // Alert.alert("Book returned to the library!");
   }
  });
};
```

Criar o método getBookDetails

```
getBookDetails = bookId => {
 bookId = bookId.trim();
 db.collection("books")
  .where("book_id", "==", bookId)
  .get()
  .then(snapshot => {
   snapshot.docs.map(doc => {
    this.setState({
     bookName: doc.data().book_details.book_name
    });
   });
  }); };
```

Criar o metodo getStudentDetails

```
getStudentDetails = studentId => {
  studentId = studentId.trim();
  db.collection("students")
   .where("student_id", "==", studentId)
   .get()
   .then(snapshot => {
    snapshot.docs.map(doc => {
     this.setState({
      studentName: doc.data().student_details.student_name
     });
    });
   });
 };
```

Implementar o método initiateBookIssue.

```
initiateBookIssue = async (bookId, studentId, bookName, studentName) => {
  //adicionar uma transação
  db.collection("transactions").add({
   student_id: studentId,
   student_name: studentName,
   book_id: bookId,
   book_name: bookName,
   date: firebase.firestore.Timestamp.now().toDate(),
   transaction_type: "issue"
  });
  //alterar status do livro
  db.collection("books")
   .doc(bookId)
   .update({
    is_book_available: false
   });
  // alterar o número de livros retirados pelo aluno
  db.collection("students")
   .doc(studentId)
   .update({
    number_of_books_issued: firebase.firestore.FieldValue.increment(1)
   });

  // atualizando estado local
  this.setState({
   bookId: "",
   studentId: ""
  });
 };
```

Implementar o método initiateBookReturn

```
initiateBookReturn = async (bookId, studentId, bookName, studentName) => {
  // adicionar uma transação
  db.collection("transactions").add({
    student_id: studentId,
    student_name: studentName,
    book_id: bookId,
    book_name: bookName,
    date: firebase.firestore.Timestamp.now().toDate(),
    transaction_type: "return"
  });
  // alterar status do livro
  db.collection("books")
    .doc(bookId)
    .update({
     is_book_available: true
    });
  // alterar o número de livros retirados pelo aluno
  db.collection("students")
    .doc(studentId)
    .update({
     number_of_books_issued: firebase.firestore.FieldValue.increment(-1)
    });

  // atualizando estado local
  this.setState({
    bookId: "",
    studentId: ""
  });
 };
```

Modificar o return.

```
return (
   <KeyboardAvoidingView behavior="padding" style={styles.container}>
    <ImageBackground source={bgImage} style={styles.bgImage}>
     <View style={styles.upperContainer}>
      <Image source={appIcon} style={styles.appIcon} />
      <Image source={appName} style={styles.appName} />
     </View>
     <View style={styles.lowerContainer}>
      <View style={styles.textinputContainer}>
       <TextInput
        style={styles.textinput}
        placeholder={"ID do Livro"}
        placeholderTextColor={"#FFFFFF"}
        value={bookId}
        onChangeText={text => this.setState({ bookId: text })}
       />
       <TouchableOpacity
        style={styles.scanbutton}
        onPress={() => this.getCameraPermissions("bookId")}
       >
        <Text style={styles.scanbuttonText}>Digitalizar</Text>
       </TouchableOpacity>
      </View>
      <View style={[styles.textinputContainer, { marginTop: 25 }]}>
       <TextInput
        style={styles.textinput}
        placeholder={"ID do Aluno"}
        placeholderTextColor={"#FFFFFF"}
        value={studentId}
        onChangeText={text => this.setState({ studentId: text })}
       />
       <TouchableOpacity
        style={styles.scanbutton}
        onPress={() => this.getCameraPermissions("studentId")}
       >
        <Text style={styles.scanbuttonText}>Digitalizar</Text>
       </TouchableOpacity>
      </View>
      <TouchableOpacity
       style={[styles.button, { marginTop: 25 }]}
       onPress={this.handleTransaction}
      >
       <Text style={styles.buttonText}>Enviar</Text>
      </TouchableOpacity>
     </View>
    </ImageBackground>
   </KeyboardAvoidingView>
  );
 }
```

```
}

const styles = StyleSheet.create({
 container: {
  flex: 1,
  backgroundColor: "#FFFFFF"
 },
 bgImage: {
  flex: 1,
  resizeMode: "cover",
  justifyContent: "center"
 },
 upperContainer: {
  flex: 0.5,
  justifyContent: "center",
  alignItems: "center"
 },
 appIcon: {
  width: 200,
  height: 200,
  resizeMode: "contain",
  marginTop: 80
 },
 appName: {
  width: 180,
  resizeMode: "contain"
 },
 lowerContainer: {
  flex: 0.5,
  alignItems: "center"
 },
 textinputContainer: {
  borderWidth: 2,
  borderRadius: 10,
  flexDirection: "row",
  backgroundColor: "#9DFD24",
  borderColor: "#FFFFFF"
 },
 textinput: {
  width: "57%",
  height: 50,
  padding: 10,
  borderColor: "#FFFFFF",
  borderRadius: 10,
  borderWidth: 3,
  fontSize: 18,
  backgroundColor: "#5653D4",
  fontFamily: "Rajdhani_600SemiBold",
  color: "#FFFFFF"
 },
 scanbutton: {
```

```
  width: 100,
  height: 50,
  backgroundColor: "#9DFD24",
  borderTopRightRadius: 10,
  borderBottomRightRadius: 10,
  justifyContent: "center",
  alignItems: "center"
 },
 scanbuttonText: {
  fontSize: 20,
  color: "#0A0101",
  fontFamily: "Rajdhani_600SemiBold"
 },
 button: {
  width: "43%",
  height: 55,
  justifyContent: "center",
  alignItems: "center",
  backgroundColor: "#F48D20",
  borderRadius: 15
 },
 buttonText: {
  fontSize: 24,
  color: "#FFFFFF",
  fontFamily: "Rajdhani_600SemiBold"
 }
});
```