Almacenado de Tareas

Necesitaremos de react-native async storage (Ver documentacion en npm) y lo instalamos

npm i @react-native-async-storage/async-storage

Esto modifica código nativo por lo cual volvemos a ejecutar npm run Android para que genere un nuevo apk virtual para que encuentre la dependencia

Importamos la dependenci

useEffect permite ejecutar una accion cuando el componente se esta lanzando

Copiamos la sección de storing que guarda la lista y el Reading para obtener la lista

Todo se va a guardar en la storeData

import React,{useState,useEffect} from "react";

import { View,Text,TextInput, TouchableOpacity,FlatList, \_View} from 'react-native'

import styles from "./Style";

import RenderItem from "./RenderItem";

import AsyncStorage from '@react-native-async-storage/async-storage';

const tasks=[

]

export interface Task {

  title:string,

  done:boolean,

  date:Date

}

export default function App(){

const [text,setText]=useState('')

const[tasks,setTasks]=useState<Task[]>([])

const storeData = async (value:Task[]) => {

  try {

    await AsyncStorage.setItem('my-todo', JSON.stringify(value));//transforma de JSON

  } catch (e) {

    // saving error

  }

}

const getData = async () => {

  try {

    const value = await AsyncStorage.getItem('my-todo');//clave no se puede llamar igual el my

    if (value !== null) {

      const tasksLocal=JSON.parse(value)//tranforma a JSON

      setTasks(tasksLocal)

    }

  } catch (e) {

    // error reading value

  }

}

useEffect(()=>{

  getData()

},[])

  const addTask=()=>{

    const tmp=[...tasks]

    const newTaks={

      title:text,

      done:false,

      date:new Date()

    }

    tmp.push(newTaks)

    setTasks(tmp)

    storeData(tmp)

    setText('')

  }

  const markDone=(task:Task)=>{

    const tmp=[...tasks]

    const index=tmp.findIndex(el=>el.title===task.title)

    const todo=tmp[index]

    todo.done=!todo.done

    setTasks(tmp)

    storeData(tmp)

  }

  const deleteFunction=(task:Task)=>{

    const tmp=[...tasks]

    const index=tmp.findIndex(el=>el.title===task.title)

    tmp.splice(index,1)//splice permite quitar un elemento dado el index y la cantidad

    setTasks(tmp)

    storeData(tmp)

  }

  return(

    <View style={styles.container}>

       <Text style={styles.title}>

          Mis tareas por hacer

       </Text>

       <View style={styles.inputconteiner}>

        <TextInput placeholder="Agregar una nueva tarea"

        style={styles.textinput}

        value={text}

        onChangeText={(t:string)=>setText(t)}

        />

        <TouchableOpacity

        onPress={addTask}

        style={styles.addButon}>

          <Text style={styles.whitetext}>

            Agregar

          </Text>

        </TouchableOpacity>

       </View>

       <View style={styles.scrollContainer}>

        <FlatList

          renderItem={({item})=>(

          <RenderItem

          item={item}

          markDone={markDone}

          deleteFuntion={deleteFunction}

          />

          )}

         data={tasks}

        />

       </View>

    </View>

  )

}

Aca sale un error debido a que con stringyfi lo transforma todo a texto hasta el date en el RenderItem necesitamos modificar la refundición del dato

import React from "react";

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

import styles from "./Style";

import { Task } from "./App";

interface ItemProps{

    item: Task;

    markDone:(task:Task)=>void;

    deleteFuntion:(task:Task)=>void;

}

export default function RenderItem({item, markDone,deleteFuntion}:ItemProps){

    return (

    <View style={styles.itemContainer}>

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

        <Text style={item.done ? styles.textDone: styles.text}>{item.title}</Text>

        <Text style={styles.text}>{new Date(item.date).toDateString()}</Text>

      </TouchableOpacity>

      {

        item.done &&

      (<TouchableOpacity style={styles.removeButtom} onPress={()=>deleteFuntion(item)}>

          <Text style={styles.whitetext}>

              Eliminar

            </Text>

        </TouchableOpacity>)

      }

    </View>

    )

  }