Análise de Sistemas App Tasks - Banco de Dados e animações

Vamos refatorar o app para que as tarefas sejam salvas utilizando o banco de dados sqlite.

- Instale a biblioteca npx expo install expo-sqlite
- Em src/types/Task.ts altere a model da interface Task para que o id seja number(vamos usar o auto incremento do banco) e completed seja number(sqlite não possui tipo booleano).
- Em src/components/ItemCard.tsx altere a interface para que handleRemoveTask e handleDoneTask tenham parâmetros do tipo number.
- Em src/screens/Home.tsx:
    - Insira: import * as SQLite from "expo-sqlite";
    - No topo, após os states, crie uma função openDatabase que deve ter o valor: const db = SQLite.openDatabase("db.db"); e return db;
    - Logo abaixo insira: const db = openDatabase();
    - Altere a função getTask para recuperar as tarefas do bd:

```
const getTasks = async () => {
    db.transaction((tx) => {
      tx.executeSql(
        `select * from tasks where completed = 0;`,
        [],
        (_, { rows: { _array } }) => {
          setTaskList(_array);
        }
      );
    });
  };
```

    - Crie duas novas funções para recuperar as tarefas por categoria e as concluídas:

```
const getTasksByCategory = (category: string) => {
  db.transaction((tx) => {
    tx.executeSql(
      `select * from tasks where completed = 0 and category = ?;`,
      [category],
      (_, { rows: { _array } }) => {
        setTaskList(_array);
      }
    );
  });
};

const getCompletedTasks = () => {
  db.transaction((tx) => {
    tx.executeSql(
      `select * from tasks where completed = 1;`,
      [],
      (_, { rows: { _array } }) => {
        setTaskList(_array);
      }
    );
  });
```

- Altere as funções de adicionar, remover e concluir:

```
const handleAddTask = async () => {
    if (taskInput !== "" && categoryValue) {
      db.transaction((tx) => {
        tx.executeSql(
          "insert into tasks (completed, title, category) values (0, ?, ?)",
          [taskInput, categoryValue]
        );
        tx.executeSql(
          `select * from tasks where completed = 0;`,
          [],
          (_, { rows: { _array } }) => {
            setTaskList(_array);
          }
        );
      });
    }

    setTaskInput("");
    setCategoryValue(null);
  };

  const handleRemoveTask = (id: number) => {
    db.transaction((tx) => {
      tx.executeSql("delete from tasks where id = ?", [id]);
      tx.executeSql(
        `select * from tasks where completed = 0;`,
        [],
        (_, { rows: { _array } }) => {
          setTaskList(_array);
        }
      );
    });
  };

  const handleDoneTask = (id: number) => {
    db.transaction((tx) => {
      tx.executeSql("update tasks set completed = ? where id = ? ", [1, id]);
      tx.executeSql(
        `select * from tasks where completed = 0;`,
        [],
        (_, { rows: { _array } }) => {
          setTaskList(_array);
        }
      );
    });
  };
```

- Remova a função storeTasks e a getTasks e o state filteredTasks e setFilteredTasks..
- Altere a função handleSelectCategory para no case ‘all’ chamar getTasks(), case ‘done’ chamar getCompletedTasks() e default chamar getTasksByCategory(type).
- Altere o useEffect para primeiro criar a tabela e depois chamar getTasks:

```
useEffect(() => {
  db.transaction((tx) => {
    tx.executeSql(
      "create table if not exists tasks (id integer primary key not null, completed int, title text, category text);"
    );
  });
  getTasks();
}, []);
```

    - Na flatlist altere os valores de filteredTasks para taskList.

- Agora vamos inserir animações:
    - Faça a importação de: import Animated, BounceInDown,  FlipInYRight, FlipOutYRight } from "react-native-reanimated";
    - Altere a FlatList das categorias para Animated.FlatList e Insira: entering={BounceInDown}.
    - Altere a FlatList das tarefas para Animated.FlatList e insira: entering={FlipInYRight} e exiting={FlipOutYRight}.
    - Altere a View da mensagem sem tarefas para Animated.View e insira: entering={BounceInDown}.
    - Acesse a documentação para explorar os tipos de animações: https://docs.swmansion.com/react-native-reanimated/docs/layout-animations/entering-exiting-animations
- Remova as importações não utilizadas.