Exercício Prático: App "Diário de Viagens"

Descrição do Projeto

Sua tarefa é criar um "Diário de Viagens" simples. O aplicativo permitirá que o usuário registre memórias de suas viagens, que serão exibidas na tela principal. Este projeto tem como foco principal a comunicação entre duas telas: a tela principal que exibe o diário e uma tela de formulário para adicionar novas entradas. O desafio é fazer com que a nova entrada, criada na segunda tela, seja enviada de volta e apareça na lista da tela principal.

Fluxo do App:

1. A tela principal exibe as entradas do diário já salvas, cada uma em um Card estilizado.
2. Um FloatingActionButton (botão +) abre uma segunda tela para criar uma "Nova Entrada".
3. A tela de "Nova Entrada" tem campos para "Título" e "Descrição" da viagem.
4. Ao clicar em "Salvar", o aplicativo **volta para a tela principal, trazendo os dados da nova entrada**. A lista na tela principal é então atualizada instantaneamente.

Código Base para os Alunos

Instruções: Copie o código abaixo para o seu arquivo main.dart. Sua tarefa é completar as seções marcadas com // TODO: .... O foco é implementar a navegação entre as telas e a lógica para passar os dados de volta da tela de formulário para a tela principal, atualizando o estado.

Dart

```
import 'package:flutter/material.dart';
import 'package:intl/intl.dart'; // Pacote para formatar a data

// --- MODELO DE DADOS ---
class EntradaDiario {
 final String titulo;
 final String descricao;
 final DateTime data;

 EntradaDiario({required this.titulo, required this.descricao, required this.data});
}

void main() {
 runApp(const MyApp());
```

```
}

class MyApp extends StatelessWidget {
 const MyApp({super.key});

 @override
 Widget build(BuildContext context) {
  return MaterialApp(
   title: 'Diário de Viagens',
   debugShowCheckedModeBanner: false,
   theme: ThemeData(primarySwatch: Colors.brown),
   home: const TelaDiario(),
  );
 }
}

// ------------------- TELA PRINCIPAL (DIÁRIO) -------------------
class TelaDiario extends StatefulWidget {
 const TelaDiario({super.key});

 @override
 State<TelaDiario> createState() => _TelaDiarioState();
}

class _TelaDiarioState extends State<TelaDiario> {
 // Nossa fonte de dados: uma lista de entradas do diário.
 final List<EntradaDiario> _entradas = [];

 // Função assíncrona para navegar para a tela de nova entrada.
 void _navegarParaNovaEntrada() async {
  // TODO: ETAPA 1 - NAVEGAR E ESPERAR O RESULTADO
```

```
  // 1. Use `Navigator.push` para navegar para a `TelaNovaEntrada`.
  // 2. A chamada a `Navigator.push` retorna um `Future`. Use a palavra-chave `await`
  //    para esperar que a tela `TelaNovaEntrada` seja fechada e retorne um dado.
  // 3. Armazene o resultado em uma variável final (ex: `final novaEntrada = await ...`).
  // 4. Verifique se a `novaEntrada` não é nula (o usuário pode ter voltado sem salvar).
  // 5. Se não for nula, use `setState` para adicionar a `novaEntrada` à nossa lista `_entradas`.



}


 @override
 Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(title: const Text('Meu Diário de Viagens')),
    body: ListView.builder(
     // TODO: ETAPA 2 - CONECTAR A LISTA À UI
     // 1. Defina o `itemCount` para ser o tamanho da sua lista `_entradas`.
     // 2. No `itemBuilder`, acesse a entrada atual usando o `index` (ex: `final entrada =
_entradas[index];`).
     // 3. Conecte as propriedades `entrada.titulo`, `entrada.descricao` e `entrada.data`
     //    aos widgets `Text` correspondentes dentro do `Card`.
     //    (Dica: para formatar a data, use `DateFormat('dd/MM/yyyy').format(entrada.data)`)

     itemCount: 0, // Substitua pelo valor correto
     itemBuilder: (context, index) {
       return Card(
         margin: const EdgeInsets.all(10.0),
         child: Padding(
           padding: const EdgeInsets.all(15.0),
           child: Column(
             crossAxisAlignment: CrossAxisAlignment.start,
```

```
            children: [
              Text(
                'Título da Viagem Aqui', // Substitua pelo título da entrada
                style: const TextStyle(fontSize: 18, fontWeight: FontWeight.bold),
              ),
              const SizedBox(height: 5),
              Text(
                'Data Aqui', // Substitua pela data formatada
                style: const TextStyle(color: Colors.grey, fontStyle: FontStyle.italic),
              ),
              const Divider(height: 20),
              Text(
                'Descrição da viagem aqui...', // Substitua pela descrição da entrada
                style: const TextStyle(fontSize: 16),
              ),
            ],
          ),
        ),
      );
    },
  ),
  floatingActionButton: FloatingActionButton(
    onPressed: _navegarParaNovaEntrada, // O botão já está conectado à função!
    child: const Icon(Icons.add),
  ),
 );
 }
}

// ------------------- TELA DE NOVA ENTRADA (FORMULÁRIO) -------------------
class TelaNovaEntrada extends StatefulWidget {
```

```
 const TelaNovaEntrada({super.key});

 @override
 State<TelaNovaEntrada> createState() => _TelaNovaEntradaState();
}

class _TelaNovaEntradaState extends State<TelaNovaEntrada> {
 final _tituloController = TextEditingController();
 final _descricaoController = TextEditingController();

 void _salvarEntrada() {
  // TODO: ETAPA 3 - CRIAR E RETORNAR OS DADOS
  // 1. Verifique se os campos de título e descrição não estão vazios.
  // 2. Se não estiverem vazios, crie uma nova instância da classe `EntradaDiario`
  //    com os dados dos controllers e a data atual (`DateTime.now()`).
  // 3. Use `Navigator.pop(context, ...)` para fechar esta tela e enviar o
  //    novo objeto `EntradaDiario` de volta para a `TelaDiario`.

 }

 @override
 Widget build(BuildContext context) {
  return Scaffold(
   appBar: AppBar(title: const Text('Nova Entrada no Diário')),
   body: Padding(
    padding: const EdgeInsets.all(20.0),
    child: Column(
     children: [
      TextField(controller: _tituloController, decoration: const InputDecoration(labelText: 'Título')),
      const SizedBox(height: 10),
```

```
        TextField(controller: _descricaoController, decoration: const InputDecoration(labelText:
'Descrição')),
        const SizedBox(height: 20),
        ElevatedButton(
          onPressed: _salvarEntrada,
          child: const Text('Salvar'),
        ),
       ],
      ),
    ),
  );
 }
}
```

Nova Biblioteca: 'package:intl/intl.dart' para formatação da data.

Nota: Para o formatador de data (DateFormat) funcionar, os alunos têm duas opções:

1. Executar o comando **flutter pub add intl** no terminal de comados do VSCode (modo automático)
2. Adicionar a dependência intl ao pubspec.yaml (modo manual):

YAML

```
dependencies:
 flutter:
  sdk: flutter
 intl: ^0.18.0 # Ou a versão mais recente
```

RECOMENDAÇÃO

TENTE IMPLEMNETAR A CODIFICAÇÃO, POREM, CASO TENHA DIFICULDADES, SEGUE ABAIXO OS CÓDIGOS DE REFERÊNCIA DAS ETAPAS.

Resolução das Etapas (Referência do Aluno)

Aqui estão os trechos de código corretos para cada etapa TODO. Use-os para corrigir ou comparar com sua implementação.

Etapa 1: Navegar e Esperar o Resultado (em _TelaDiarioState)

Dart

```
void _navegarParaNovaEntrada() async {
 final novaEntrada = await Navigator.push(
  context,
  MaterialPageRoute(builder: (context) => const TelaNovaEntrada()),
 );


 if (novaEntrada != null) {
  setState(() {
   _entradas.add(novaEntrada as EntradaDiario);
  });
 }
}
```

Etapa 2: Conectar a Lista à UI (em _TelaDiarioState)

Dart

```
// ...
body: ListView.builder(
 itemCount: _entradas.length,
 itemBuilder: (context, index) {
  final entrada = _entradas[index];
  return Card(
   margin: const EdgeInsets.all(10.0),
   child: Padding(
    padding: const EdgeInsets.all(15.0),
    child: Column(
     crossAxisAlignment: CrossAxisAlignment.start,
     children: [
```

```
        Text(
          entrada.titulo,
          style: const TextStyle(fontSize: 18, fontWeight: FontWeight.bold),
        ),
        const SizedBox(height: 5),
        Text(
          DateFormat('dd/MM/yyyy').format(entrada.data),
          style: const TextStyle(color: Colors.grey, fontStyle: FontStyle.italic),
        ),
        const Divider(height: 20),
        Text(
          entrada.descricao,
          style: const TextStyle(fontSize: 16),
        ),
       ],
      ),
    ),
   );
 },
),
// ...
```

Etapa 3: Criar e Retornar os Dados (em _TelaNovaEntradaState)

Dart

```
void _salvarEntrada() {
 if (_tituloController.text.isNotEmpty && _descricaoController.text.isNotEmpty) {
  final novaEntrada = EntradaDiario(
   titulo: _tituloController.text,
   descricao: _descricaoController.text,
   data: DateTime.now(),
  );
  Navigator.pop(context, novaEntrada);
```

```
 }

}
```
