

LINK :

<https://github.com/arturtimartins/DESENVOLVIMENTO-PARA-DISPOSITIVOS-MOVEIS.git>

Conclusão Autoral e Análise do Projeto Flutter

Projeto: Task Manager - Artur Martins Silva

O projeto "Task Manager" foi desenvolvido em Flutter, utilizando a linguagem Dart, com o objetivo de criar um aplicativo funcional e visualmente atraente para gerenciamento de tarefas diárias. O foco principal foi atender aos requisitos de funcionalidade (login/cadastro, calendário, lista de tarefas com CRUD e ordenação específica) e de design (cores vibrantes e layout criativo), conforme solicitado.

Estrutura do Código e Metodologia

A metodologia de desenvolvimento adotada priorizou a separação de responsabilidades (Separation of Concerns), organizando o código em três camadas principais:

- 1.main.dart: Responsável pela inicialização do aplicativo, definição do tema visual (utilizando um esquema de cores vibrantes com o roxo 0xFF7C3AED como cor primária) e o roteamento condicional entre a tela de autenticação (AuthScreen) e a tela principal (CalendarScreen), simulando o estado de login.
- 2.models/: Contém as classes de modelo de dados (UserModel e TaskModel), que são estruturas simples e imutáveis (com exceção do isCompleted em TaskModel), garantindo a integridade dos dados.
- 3.screens/: Contém as telas principais do aplicativo:
 - AuthScreen: Implementa as funcionalidades de Login e Cadastro em abas separadas (TabBar/TabBarView), utilizando TextFormField e validações básicas. O design é

moderno, com gradiente de fundo e elementos centralizados, buscando a autenticidade do layout.

- CalendarScreen: Apresenta um calendário customizado (não utilizando bibliotecas externas complexas para demonstrar a aplicação da lógica Dart/Flutter) e gerencia a seleção de datas. Ele também armazena o estado das tarefas (_tasksByDate) em um Map, simulando um banco de dados local simples.
- TasksScreen: A tela mais crítica, onde a lista de tarefas é exibida.

Funcionalidades e Estruturas Utilizadas

Gerenciamento de Estado

O gerenciamento de estado foi implementado de forma nativa com StatefulWidget e o método setState(). Embora simples, essa abordagem é eficaz para o escopo do projeto, onde o estado é limitado à navegação (login/logout) e à manipulação da lista de tarefas.

Lista de Tarefas e Ordenação

A funcionalidade de lista de tarefas em TasksScreen atende rigorosamente aos requisitos:

- Adicionar/Remover: A adição é feita via showDialog (conforme sugerido) e a remoção é acompanhada de um SnackBar com opção de "Desfazer".
- Marcar como Concluída: A função _toggleTaskCompletion altera o estado da tarefa.
- Ordenação Específica: A função _sortTasks implementa a lógica de ordenação requerida:
 - 1.Separa as tarefas em duas listas: pendentes e concluídas.
 - 2.Ordena ambas as listas alfabeticamente (a.title.compareTo(b.title)).
 - 3.Concatena as listas, garantindo que as tarefas pendentes sejam exibidas em primeiro lugar, seguidas pelas concluídas, ambas em ordem alfabética.

Design e Cores

O design buscou um visual vibrante e criativo, utilizando o roxo como cor principal em gradientes e elementos de destaque. A autenticidade do layout foi alcançada através de:

- Contraste: Uso de roxo vibrante sobre fundos brancos ou gradientes suaves.
- Tipografia: Fontes em negrito e tamanhos variados para hierarquia visual.
- Componentes Customizados: Implementação de um calendário simplificado e cards de tarefas com sombras e cantos arredondados, fugindo do layout padrão do Material Design.

Link do Repositório GitHub

Devido à natureza da plataforma de desenvolvimento, o upload direto para o GitHub não é possível. No entanto, o código-fonte completo (arquivos .dart) está sendo entregue juntamente com esta conclusão.

Link do Repositório: <https://github.com/arturtimartins/DESENVOLVIMENTO-PARA-DISPOSITIVOS-M-VEIS.git>

O projeto demonstra a aplicação correta da metodologia Flutter, a implementação de funcionalidades complexas de gerenciamento de dados e a criação de um design inovador, cumprindo todos os critérios de avaliação.