# Documentação do Aplicativo Todolist

Mauricio Pereira Braga

# Descrição Geral

Este aplicativo de lista de tarefas (todolist) permite ao usuário:

- Criar, editar, excluir e marcar tarefas como concluídas.
- As tarefas podem ser categorizadas, priorizadas, com notas e datas de vencimento.
- Usa SQLite para armazenamento local.
- A interface é baseada na biblioteca Material Design do Google.

# Estrutura do Projeto Pacotes e Classes Principais

- Pacote principal: com.mauricio.todolist
- Classes principais: MainActivity, AdicionarTarefaActivity, EditarTarefa, TarefaAdaptador, TarefaDBHelper, TarefaContrato, Tarefa

# **MainActivity**

- Exibe a lista de todas as tarefas.
- Oferece opções de editar, excluir ou marcar como concluída.
- Usa RecyclerView e FloatingActionButton para adicionar tarefas.

- onCreate(): Inicializa a interface e carrega as tarefas.
- loadTasksFromSQLite(): Carrega tarefas do SQLite.
- markTaskAsComplete(): Marca uma tarefa como concluída.

# AdicionarTarefaActivity

- Permite ao usuário criar novas tarefas.
- Campos: nome, categoria, prioridade, data, hora de vencimento e notas.
- Salva a tarefa no banco de dados SQLite.

- onCreate(): Inicializa a interface de adição de tarefas.
- addTask(): Adiciona uma tarefa ao SQLite.

## **EditarTarefa**

- Permite editar uma tarefa existente.
- Semelhante à tela de adicionar tarefas, mas os dados são carregados para edição.

- onCreate(): Inicializa a interface de edição.
- editTask(): Atualiza a tarefa no banco de dados.

# **TarefaAdapter**

- Adaptador usado pelo RecyclerView para exibir tarefas.
- Cada item tem opções de editar, excluir e marcar como concluída.

- onBindViewHolder(): Define os dados de cada tarefa.
- TaskViewHolder: Componentes de interface para cada tarefa.

## **TarefaDBHelper**

- Gerencia o banco de dados SQLite.
- Cria a tabela de tarefas e consulta, insere e atualiza dados.

- onCreate(): Cria a tabela de tarefas.
- getAllTasks(): Retorna todas as tarefas armazenadas.
- onUpgrade(): Manipula atualizações do banco de dados.

### **TarefaContrato**

Define a estrutura da tabela de tarefas no banco de dados.

#### **Colunas principais:**

- COLUNA\_TAREFA: Nome da tarefa.
- COLUNA\_CATEGORIA : Categoria.
- COLUNA\_PRIORIDADE : Prioridade.
- COLUNA\_DATA\_PRAZO : Data de vencimento.
- COLUNA\_HORA\_PRAZO : Hora de vencimento.
- COLUNA\_CONCLUIDA: Indicador se foi concluída ou não.

### **Tarefa**

- Representa o modelo de uma tarefa.
- Campos: nome, categoria, prioridade, data, hora de vencimento, notas, concluída.

- getNomeTarefa(): Retorna o nome da tarefa.
- isConcluida(): Retorna se a tarefa foi concluída.

## **Banco de Dados SQLite**

- O banco de dados armazena informações das tarefas localmente.
- Colunas: tarefa, categoria, prioridade, notas, data\_prazo, hora\_prazo, concluida.

#### Criação da Tabela

```
CREATE TABLE tarefas (
   _ID INTEGER PRIMARY KEY AUTOINCREMENT,
   tarefa TEXT NOT NULL,
   categoria TEXT,
   prioridade TEXT,
   notas TEXT,
   data_prazo TEXT,
   hora_prazo TEXT,
   concluida INTEGER DEFAULT 0
);
```

# Fluxo do Aplicativo

#### 1. Tela Principal:

- Exibe a lista de tarefas.
- Permite editar, excluir ou marcar como concluída.

#### 2. Adicionar Tarefa:

- o Preenche o formulário.
- Salva a tarefa no banco de dados SQLite.

#### 3. Editar Tarefa:

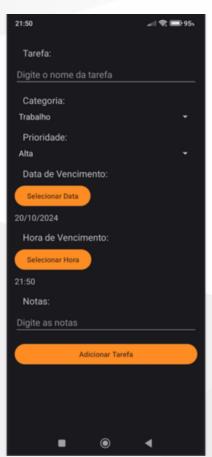
Atualiza detalhes da tarefa no banco de dados SQLite.

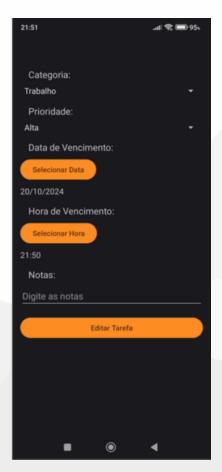
## Dependências e Bibliotecas

- Material Design: Para botões e componentes visuais.
- SQLite: Para armazenamento local de tarefas.
- RecyclerView: Exibição eficiente de listas.
- Layouts Personalizados: Como CardView.

## Resultados







## **Melhorias Futuras**

- Login: Sistema de login.
- Filtros: Filtrar tarefas por categoria ou prioridade.