MediTrack - Aplicativo de Controle de Medicamentos

Este é um aplicativo Android desenvolvido em Kotlin para ajudar usuários a registrar, acompanhar e lembrar de tomar seus medicamentos. Ele segue as melhores práticas de arquitetura e inclui testes automatizados.

Funcionalidades

- **Cadastro de Medicamentos**: Permite registrar nome, dosagem, frequência e horários.
- **Lista de Medicamentos**: Exibe todos os medicamentos cadastrados com status (pendente, tomado, atrasado).
- **Notificações de Lembrete**: Notifica o usuário quando for hora de tomar um remédio.
- **Histórico de Uso**: Mostra registros de medicamentos tomados, pendentes ou esquecidos.
- **Marcar como "Tomado"**: Permite ao usuário marcar um medicamento como tomado.
- **Exportar Histórico (PDF)**: Geração de relatório dos medicamentos tomados.

Arquitetura

O aplicativo foi desenvolvido seguindo a arquitetura MVVM (Model-View-ViewModel) com o padrão Repository. Utiliza as seguintes tecnologias e bibliotecas:

- **Kotlin**: Linguagem de programação principal.
- **Jetpack Compose**: Para construção da interface do usuário.
- **Room Persistence Library**: Para banco de dados local.
- **WorkManager**: Para agendamento de notificações de lembrete.
- **Navigation Compose**: Para navegação entre as telas.

- **OpenPDF**: Para geração de relatórios em PDF.

Estrutura do Projeto

```
MediTrack/
├── app/
│   ├── src/
│   │   ├── main/
│   │   │   ├── AndroidManifest.xml
│   │   │   ├── java/
│   │   │   │   └── com/
│   │   │   │       └── example/
│   │   │   │           └── meditrack/
│   │   │   │               ├── data/
│   │   │   │               │   ├── dao/
│   │   │   │               │   │   └── MedicationDao.kt
│   │   │   │               │   ├── database/
│   │   │   │               │   │   └── MedicationDatabase.kt
│   │   │   │               │   └── model/
│   │   │   │               │       └── Medication.kt
│   │   │   │               ├── di/
│   │   │   │               │   └── AppContainer.kt
│   │   │   │               ├── ui/
│   │   │   │               │   ├── screens/
│   │   │   │               │   │   ├── AddEditMedicationScreen.kt
│   │   │   │               │   │   ├── MedicationDetailScreen.kt
│   │   │   │               │   │   └── MedicationListScreen.kt
│   │   │   │               │   └── viewmodel/
│   │   │   │               │       └── MedicationViewModel.kt
│   │   │   │               ├── util/
│   │   │   │               │   ├── NotificationScheduler.kt
│   │   │   │               │   └── PdfGenerator.kt
│   │   │   │               ├── worker/
│   │   │   │               │   └── NotificationWorker.kt
│   │   │   │               ├── MainActivity.kt
│   │   │   │               └── MediTrackApplication.kt
│   │   │   └── res/
│   │   │       ├── drawable/
│   │   │       │   └── ic_launcher_foreground.xml
│   │   │       └── values/
│   │   │           └── strings.xml
│   │   ├── androidTest/
│   │   │   └── java/
│   │   │       └── com/
│   │   │           └── example/
│   │   │               └── meditrack/
│   │   │                   ├── data/
│   │   │                   │   └── dao/
│   │   │                   │       └── MedicationDaoTest.kt
│   │   │                   └── ui/
│   │   │                       └── MedicationUiTest.kt
│   │   └── test/
│   │       └── java/
│   │           └── com/
│   │               └── example/
│   │                   └── meditrack/
│   │                       └── data/
│   │                           └── model/
│   │                               └── MedicationTest.kt
│   └── build.gradle.kts
├── build.gradle.kts
└── settings.gradle.kts
```

Como Rodar o Projeto

1. **Pré-requisitos**: Certifique-se de ter o Android Studio instalado e configurado em sua máquina.
2. **Clonar o Repositório**: Se este projeto estivesse em um repositório Git, você o clonaria. Como está em um ambiente sandbox, os arquivos já estão disponíveis.
3. **Abrir no Android Studio**: Abra o projeto `MediTrack` no Android Studio.
4. **Sincronizar Gradle**: O Android Studio deve sincronizar automaticamente o projeto com os arquivos Gradle. Se não, clique em "Sync Project with Gradle Files" na barra de ferramentas.
5. **Executar no Emulador/Dispositivo**: Selecione um emulador Android ou conecte um dispositivo físico e clique no botão "Run" (ícone de play verde) no Android Studio para instalar e executar o aplicativo.

Testes Automatizados

Testes Unitários

Os testes unitários estão localizados em `app/src/test/java/`. Eles verificam a lógica de negócios e validações de forma isolada. Exemplo:

`app/src/test/java/com/example/meditrack/data/model/MedicationTest.kt`

Testes de DAO (Room in-memory database)

Os testes de DAO estão localizados em `app/src/androidTest/java/`. Eles testam a interação com o banco de dados Room usando um banco de dados em memória para agilidade. Exemplo:

`app/src/androidTest/java/com/example/meditrack/data/dao/MedicationDaoTest.kt`

Testes de UI (Espresso/Compose Test)

Os testes de UI estão localizados em `app/src/androidTest/java/`. Eles simulam interações do usuário com a interface do aplicativo. Exemplo:

`app/src/androidTest/java/com/example/meditrack/ui/MedicationUiTest.kt`

Para executar os testes, você pode usar o Android Studio ou a linha de comando Gradle:

- **Android Studio**: Clique com o botão direito no diretório `test` ou `androidTest` e selecione "Run Tests".
- **Linha de Comando**: Navegue até o diretório raiz do projeto (`MediTrack/`) e execute: `bash ./gradlew test ./gradlew connectedCheck`

Próximos Passos e Melhorias

- Implementar a funcionalidade de histórico de uso mais detalhada.
- Adicionar opção de exportar para CSV.
- Melhorar a interface do usuário com mais elementos de design e animações.
- Adicionar validação de entrada de usuário mais robusta nas telas de cadastro/edição.
- Implementar testes de integração mais abrangentes.
- Considerar a internacionalização (i18n) do aplicativo.
- Adicionar autenticação de usuário (se necessário para futuras funcionalidades).
