# Validação da Simulação de Backend - App de Doações

## Resumo da Implementação

O aplicativo **App de Doações** implementa uma simulação completa de backend através de **Data Sources Mock** que replicam fielmente o comportamento de APIs reais. Esta abordagem foi escolhida para demonstrar a arquitetura e funcionalidades sem depender de serviços externos.

## 🎯 Componentes de Simulação Implementados

### 1. InstituicaoDataSource

**Localização**: app/src/main/java/com/doacoes/data/datasource/InstituicaoDataSource.kt

**Funcionalidades Simuladas**:

* ✅ **API REST simulada** com dados de 3 instituições completas
* ✅ **Latência de rede realista** (1 segundo de delay)
* ✅ **Dados estruturados** com todas as informações necessárias:
  + Nome, descrição, categoria
  + CNPJ, telefone, email, website
  + URLs de imagens (Unsplash) para carregamento real
* ✅ **Ordenação por nome** para consistência

**Instituições Mock Implementadas**:

1. Casa de Apoio à Criança Carente (Crianças/Adolescentes)
2. Lar dos Idosos São Vicente (Idosos)
3. ONG Proteção Animal (Animais)

### 2. CieloLioDataSource

**Localização**: app/src/main/java/com/doacoes/data/datasource/CieloLioDataSource.kt

**Funcionalidades Simuladas**:

* ✅ **Fluxo completo de pagamento** com 3 estados:
  + Processando: Estado inicial com loading
  + Sucesso: Transação aprovada (90% dos casos)
  + Erro: Falha na transação (10% dos casos)
* ✅ **Tempo de processamento realista**: 2-3 segundos
* ✅ **Taxa de sucesso configurável**: 90% (simulando cenário real)
* ✅ **Geração de comprovante textual** com:
  + ID da transação único
  + Data/hora da transação
  + Descrição e valor
  + Status e meio de pagamento simulado
* ✅ **Tratamento de erros variados**:
  + "Cartão recusado pela operadora"
  + "Saldo insuficiente"
  + "Erro de comunicação com o banco"

## 🧪 Validações de Negócio Implementadas

### ProcessarDoacaoUseCase

**Localização**: app/src/main/java/com/doacoes/domain/usecase/ProcessarDoacaoUseCase.kt

**Regras Implementadas**:

* ✅ **Valor mínimo**: R$ 5,00 (constante configurável)
* ✅ **Validação de valor positivo**: Rejeita valores zero ou negativos
* ✅ **Geração de referência única**: Formato "DOA\_timestamp"
* ✅ **Tratamento de exceções**: Mensagens claras para o usuário

## 📱 Integração com a UI

### Estados Visuais Implementados

* ✅ **Loading State**: Indicador de progresso durante carregamento de instituições
* ✅ **Processing State**: Feedback visual durante processamento de pagamento
* ✅ **Success State**: Confirmação visual com detalhes da transação
* ✅ **Error State**: Mensagens de erro com opção "Tentar Novamente"
* ✅ **Empty State**: Tratamento para lista vazia de instituições

### Fluxo de Dados Reativo

* ✅ **StateFlow/Flow**: Streams reativos para todos os dados
* ✅ **Compose State**: UI reativa que responde automaticamente às mudanças
* ✅ **Error Handling**: Propagação e tratamento adequado de erros

## 🔍 Testes de Validação

### Testes Unitários Implementados

**Localização**: app/src/test/java/com/doacoes/domain/usecase/

1. **GetInstituicoesUseCaseTest**:
   * ✅ Testa retorno correto da lista de instituições
   * ✅ Testa cenário de lista vazia
   * ✅ Usa mocks para isolar a lógica de negócio
2. **ProcessarDoacaoUseCaseTest**:
   * ✅ Testa processamento com valor válido
   * ✅ Testa validação de valor mínimo
   * ✅ Testa validação de valor zero
   * ✅ Verifica criação correta do request de pagamento

## 🎮 Como Testar a Simulação

### Cenários de Teste Disponíveis

1. **Carregamento de Instituições**:
   * Abra o app e observe o loading de 1 segundo
   * Verifique se as 3 instituições são exibidas corretamente
   * Teste o botão "Tentar Novamente" em caso de erro simulado
2. **Fluxo de Doação Completo**:
   * Selecione uma instituição
   * Escolha um valor predefinido (R$ 10, 25, 50, 100)
   * Ou insira um valor personalizado
   * Clique em "Doar Agora"
   * Observe o estado de processamento (2-3 segundos)
   * Verifique o resultado (90% sucesso, 10% erro)
3. **Validações de Valor**:
   * Tente inserir valor menor que R$ 5,00
   * Tente inserir valor zero ou negativo
   * Verifique as mensagens de erro exibidas
4. **Tratamento de Erros**:
   * Execute várias doações para ver diferentes tipos de erro
   * Teste o botão "Tentar Novamente" após erro
   * Verifique se o estado é limpo corretamente

## ✅ Checklist de Validação

* **Simulação de API REST** para instituições
* **Simulação de API Cielo Lio** para pagamentos
* **Latência de rede realista** implementada
* **Estados de loading/success/error** funcionais
* **Validações de negócio** implementadas e testadas
* **Geração de comprovantes** simulados
* **Taxa de sucesso/falha** configurável
* **Tratamento de exceções** adequado
* **Testes unitários** cobrindo cenários principais
* **UI reativa** respondendo aos estados do backend
* **Documentação completa** do comportamento simulado

## 🏆 Conclusão

A simulação de backend implementada no **App de Doações** é **completa e funcional**, demonstrando:

1. **Arquitetura robusta** com separação clara de responsabilidades
2. **Comportamento realista** que simula fielmente APIs reais
3. **Tratamento adequado** de todos os estados possíveis
4. **Validações de negócio** implementadas corretamente
5. **Testabilidade** através de mocks e testes unitários
6. **Experiência de usuário** completa com feedback visual adequado

O backend mock não apenas atende aos requisitos da prova técnica, mas demonstra competência técnica em desenvolvimento Android moderno com Clean Architecture, Jetpack Compose e melhores práticas de desenvolvimento.