# Guia Completo de Testes Django com Pytest

Este documento fornece um guia para o sistema de testes do projeto Django usando pytest.

## 🎯 Visão Geral

Este projeto utiliza pytest para testes, fornecendo:

* **Testes Unitários**: Testam componentes individuais
* **Cobertura de Código**: Relatórios detalhados de cobertura
* **Fixtures Reutilizáveis**: Para criação de dados de teste

## 📁 Estrutura dos Testes

tests/  
├── conftest.py # Configurações e fixtures globais  
├── pytest.ini # Configuração do pytest  
├── test\_models.py # Testes dos modelos Django  
├── test\_views.py # Testes das views  
├── test\_forms.py # Testes dos formulários  
├── test\_urls.py # Testes das URLs

## 🚀 Configuração e Instalação

### 1. Instalar Dependências

# Instalar dependências de teste  
pip install -r requirements-test.txt

### 2. Configurar Banco de Dados

# Criar migrações (se necessário)  
python manage.py makemigrations  
  
# Aplicar migrações  
python manage.py migrate

# Executar testes   
pytest

## 📊 Tipos de Teste

### 1. Testes de Modelos (test\_models.py)

Testam os modelos Django incluindo:

* ✅ Criação e validação de instâncias
* ✅ Relacionamentos entre modelos
* ✅ Métodos personalizados
* ✅ Validações e restrições
* ✅ Cenários de erro

### 2. Testes de Views (test\_views.py)

Testam as views da aplicação:

* ✅ Autenticação e autorização
* ✅ Respostas HTTP corretas
* ✅ Contexto das templates
* ✅ Redirecionamentos
* ✅ Tratamento de erros

### 3. Testes de Formulários (test\_forms.py)

Testam os formulários Django:

* ✅ Validação de campos
* ✅ Dados válidos e inválidos
* ✅ Classes CSS (Bootstrap)
* ✅ Segurança contra injeção
* ✅ Campos obrigatórios

### 4. Testes de URLs (test\_urls.py)

Testam as configurações de URL:

* ✅ Mapeamento correto de URLs
* ✅ Parâmetros de URL
* ✅ Resolução de nomes de URL
* ✅ URLs alternativas

### 

## 🏭 Fixtures e Factories

### Fixtures Principais (conftest.py)

@pytest.fixture  
def user(user\_factory):  
 """Usuário de teste padrão."""  
 return user\_factory()  
  
@pytest.fixture  
def account(account\_factory, user):  
 """Conta de teste padrão."""  
 return account\_factory(user=user)  
  
@pytest.fixture  
def authenticated\_client(client, user):  
 """Cliente autenticado."""  
 client.force\_login(user)  
 return client

### Factories Disponíveis

* user\_factory: Cria usuários únicos
* account\_factory: Cria contas bancárias
* stock\_factory: Cria ações
* trade\_factory: Cria operações de trade
* transfer\_factory: Cria transferências
* favorite\_stock\_factory: Cria favoritos

## 📈 Cobertura de Código

### Métricas de Cobertura

* **Meta mínima**: 50%
* **Relatórios**: HTML e terminal
* **Exclusões**: Migrações, configurações

### Interpretando Relatórios

# Visualizar cobertura no terminal  
pytest --cov=core --cov-report=term-missing  
  
# Arquivos com baixa cobertura aparecem em vermelho  
# Linhas não cobertas são listada

Autor: Matheus Maia

Versão: 0.1

Data: 2025-25-06