

Planejamento de teste - Cinema Challnege

Planejamento Técnico de Automação de Testes – Sistema Cinema

1. Introdução

Este documento tem como objetivo apresentar uma análise técnica detalhada do projeto de automação de testes desenvolvido para o sistema Cinema, contemplando a parte de funcionalidades da API no back-end e a interface visual da aplicação no front-end .

O projeto foi conduzido com base em práticas de garantia de qualidade (QA) e automação funcional, visando assegurar a estabilidade, confiabilidade e manutenibilidade da aplicação.

2. Objetivo do Projeto

O objetivo principal da automação foi:

- Garantir o funcionamento correto dos principais fluxos do sistema de reservas de cinema;
- Validar a integridade das comunicações entre o frontend e o backend;
- Reduzir o tempo de regressão através da automação de testes repetitivos;
- Prover uma base de testes automatizados escalável e de fácil manutenção.

Pessoas Envolvidas:

- Raphael Sousa Rabelo Rates

3. Escopo

Nessa seção, é explicada como as quais funcionalidades, módulos ou áreas do sistema serão cobertas pelos testes, evitando ambiguidades e garantindo foco. Mostraremos as funcionalidades principais da aplicação (ex.: cadastro de usuário, login, controle de filmes, procura de sessions, procura de reservas), tipos de testes planejados (unitários, API, performance, segurança), ambiente de testes (desenvolvimento, homologação, produção simulada).

Módulo	Tipo	Descrição	Status
Autenticação	API + UI	Login, cadastro, logout e validações de token	Concluído

Filmes	API + UI	Listagem e busca de filmes	Concluído
Sessões	API	Listagem, busca por ID e por filme	Concluído
Reservas	API + UI	Criação e validação de reservas de assentos	Concluído
Testes Negativos	API	Validação de erros e respostas inválidas	Concluído
Permissões (Admin)	API	Testes administrativos (rota instável)	Não Implementado
Cinemas (Theaters)	API	Listagem e detalhes de cinemas	Não Implementado
Usuários(Admin)	API	Controle dos usuários	Não implementado

4. Ferramentas e Tecnologias

Categoria	Ferramenta	Finalidade
Framework de automação	Robot Framework	Base estrutural dos testes
Biblioteca de API	RequestsLibrary	Comunicação HTTP e validação de endpoints
Biblioteca de UI	Browser (Playwright)	Automação de interface web

Linguagem de definição	Sintaxe Robot	Escrita dos cenários e keywords
Gerador de Dados	Robot Framework Faker	Geração de dados falso para testes de validação de I/O
Carregados de arquivos JSON	Robot Framework JSON Loader	uso de dados pré-definidos para verificação dos dados
Controle de versão	Git / GitHub	Versionamento e documentação
Linguagem de Programação	Python 3.x	para uso de keywords personalizadas
Criptografia	bcrypt	simular uma criptografia usada na API
Banco de Dados	python-mongo	usado para criar keywords que interagem com o banco

5. Estrutura do Projeto

A estrutura do repositório foi organizada em módulos independentes para facilitar a manutenção e a execução seletiva dos testes, essa arquitetura permite a execução modular de testes por categoria (API ou UI), além de garantir isolamento entre os ambientes.

projeto/

```

├── tests/
│   ├── web/
│   │   ├── auth_tests.robot
│   │   ├── profile_tests.robot
│   │   ├── reservation_tests.robot
│   │   └── ui_components_tests.robot
│   └── server/

```

```
| | |— auth.robot
| | |— theaters.robot
| | |— sessions.robot
| | |— setup.robot
| | |— reservations.robot
| | |— movies.robot
|
|— resources/
| |— base.resource
| |— variables.resource
| |— pages/
| | |— reservation_page.resource
| | |— register_page.resource
| | |— profile.resource
| | |— movies_search.resource
| |— components/
| | |— auth/
| | | |— login_form.resource
| | | |— register_form.resource
| | | |— logout_button.resource
| | |— header.resource
| | |— footer.resource
| | |— movie_card.resource
|
|— api/
| | |— rauth.resource
| | |— movies.resource
| | |— reservations.resource
| | |— sessions.resource
| | |— setup.resource
| | |— theaters.resource
```

6. Cenários Implementados (com Nomeações Oficiais)

Os cenários foram divididos em cenários de interface e de Implementação, dividindo os papéis de UI e de API do projeto.

6.1 Testes de Interface

6.1.1 Módulo de Autenticação e Páginas iniciais (UI)

ID Teste	Descrição	Prioridade	Status
TC001	Verificar Elementos da Página de Login	Alta	 Implementado
TC002	Login Bem Sucedido com Credenciais Válidas	Crítica	 Implementado
TC003	Login com Email Inválido	Alta	 Implementado
TC004	Login com Senha Inválida	Alta	 Implementado
TC005	Login com Campos Vazios	Alta	 Implementado
TC006	Navegar para Página de Registro	Média	 Implementado
TC007	Login com Email Mal Formatado	Alta	 Implementado
TC008	Login com Múltiplas Tentativas	Média	 Implementado
TC009	Verificar Elementos da Página de Registro	Alta	 Implementado
TC010	Registro Bem Sucedido com Novo Usuário	Crítica	 Implementado

TC011	Registro com Email Já Existente	Alta	✓ Implementado
TC012	Registro com Senhas Diferentes	Alta	✓ Implementado
TC013	Registro com Campos Vazios	Alta	✓ Implementado
TC014	Navegar para Página de Login	Média	✓ Implementado
TC015	Registro com Email Inválido	Alta	✓ Implementado
TC016	Registro com Senha Fraca	Alta	✓ Implementado
TC017	Registro com Nome Muito Curto	Média	✓ Implementado
TC018	Logout após o Login	Crítica	✓ Implementado
TC019	Logout após o Register	Média	✓ Implementado

6.1.2. Módulo Componentes (UI)

ID Teste	Descrição	Prioridade	Status
TC022	Verificar Header Na Página Inicial	Alta	✓ Implementado
TC023	Verificar Navegação Via Header	Crítica	✓ Implementado

TC024	Verificar Header Em Diferentes Páginas	Média	✓ Implementado
TC025	Verificar Seção De Features Na Página Inicial	Média	✓ Implementado
TC026	Verificar Footer Na Página Inicial	Média	✓ Implementado
TC027	Verificar Footer Em Diferentes Páginas	Baixa	✓ Implementado
TC028	Teste Completo Seção de Filmes	Alta	✓ Implementado
TC029	Teste Navegação Entre Filmes	Alta	✓ Implementado
TC030	Teste Componentes Da Página De Busca	Alta	✓ Implementado
TC031	Teste Busca Por Filme Existente	Crítica	✓ Implementado
TC032	Teste Filtro Por Gênero	Alta	✓ Implementado
TC033	Teste Busca E Filtro Combinados	Alta	✓ Implementado
TC034	Teste Navegação Para Detalhes Do Filme	Alta	✓ Implementado

6.1.3. Módulo de Reservas (UI)

ID Teste	Descrição	Prioridade	Status
TC050	Verificar Carregamento da Página de Reservas	Alta	 Implementado
TC051	Verificar Estrutura do Card de Reserva	Alta	 Implementado
TC052	Validar Conteúdo da Reserva do Avengers	Média	 Implementado
TC053	Verificar Ícones dos Detalhes	Baixa	 Implementado
TC054	Validar Status da Reserva	Alta	 Implementado
TC055	Verificar Informações de Pagamento	Crítica	 Implementado
TC056	Validar Botões de Ação	Média	 Implementado
TC057	Navegar para Página Inicial	Média	 Implementado
TC058	Navegar para Filmes em Cartaz	Média	 Implementado
TC059	Verificar Número da Reserva	Média	 Implementado
TC060	Validar Poster do Filme	Baixa	 Implementado

TC061	Verificar Múltiplas Reservas	Alta	✓ Implementado
TC062	Validar Responsividade do Card	Média	⚠ Parcial
TC063	Verificar Ordenação das Reservas	Baixa	⚠ Pendente
TC064	Validar Comportamento Sem Reservas	Média	⚠ Pendente

6.1.4 Resumo dos testes

Módulo	Total	Implementados	Parciais	Pendentes
Autenticação	19	19	0	0
Componentes UI	13	13	0	0
Perfil	15	15	0	0
Reservas	15	12	1	2
TOTAL	64	59	1	3

6.1.5 Tabela de criticalidade.

Prioridade	Quantidade	Percentual
Crítica	8	12.5%
Alta	33	51.6%
Média	20	31.3%

Baixa	5	7.8%
-------	---	------

6.2 Testes de Implementação (API)

6.2.1. Módulo de Autenticação (API)

ID Teste	Descrição	Prioridade	Status
TC001	Cadastro de Usuário com Email Já Existente	Alta	 Implementado
TC002	Cadastro com Dados Inválidos	Crítica	 Implementado
TC003	Login com Credenciais Inválidas	Crítica	 Implementado
TC004	Acesso ao Perfil com Token Inválido	Alta	 Implementado
TC005	Fluxo Completo de Autenticação (Registro → Login → Perfil → Update)	Crítica	 Implementado
TC006	Login com Credenciais do Raphael	Média	 Implementado

6.2.2. Testes de Movies (API)

ID Teste	Descrição	Prioridade	Status
----------	-----------	------------	--------

TC007	Get All Movies With Public Access	Alta	 Implementado
TC008	Get Movies With Title Filter	Alta	 Implementado
TC009	Get Movies With Pagination	Média	 Implementado
TC010	Get Movies With Sorting	Média	 Implementado
TC011	Create Movie As Admin	Crítica	 Implementado
TC012	Create Movie With Invalid Data	Alta	 Implementado
TC013	Get Movie By Valid ID	Alta	 Implementado
TC014	Get Movie By Invalid ID	Média	 Implementado
TC015	Get Non-Existent Movie	Média	 Implementado
TC016	Update Movie As Admin	Crítica	 Implementado
TC017	Update Movie With Partial Data	Alta	 Implementado
TC018	Update Non-Existent Movie	Média	 Implementado
TC019	Delete Movie As Admin	Alta	 Implementado
TC020	Create Movie Without Admin Rights	Crítica	 Implementado

TC021	Update Movie Without Admin Rights	Crítica	✓ Implementado
TC022	Delete Movie Without Admin Rights	Crítica	✓ Implementado
TC023	Complete Movie CRUD Flow	Crítica	✓ Implementado
TC024	Get Movies Without Authentication	Alta	✓ Implementado
TC027	Search Movies By Partial Title	Média	✓ Implementado
TC028	Filter Movies By Multiple Genres	Média	✓ Implementado

6.2.3. Testes de Reservations (API)



ID Teste	Descrição	Prioridade	Status
TC029	Get My Reservations As User	Alta	✓ Implementado
TC030	Create Reservation As User	Crítica	✓ Implementado
TC031	Create Reservation With Invalid Session	Alta	✓ Implementado
TC032	Get Reservation By ID As Owner	Alta	✓ Implementado

TC033	Get All Reservations As Admin	Alta	 Implementado
TC034	Get All Reservations With Pagination	Média	 Implementado
TC035	Update Reservation Status As Admin	Crítica	 Implementado
TC036	Update Reservation Payment Status As Admin	Crítica	 Implementado
TC037	Delete Reservation As Admin	Alta	 Implementado
TC038	Get All Reservations Without Admin Rights	Crítica	 Implementado
TC039	Update Reservation Without Admin Rights	Crítica	 Implementado
TC040	Delete Reservation Without Admin Rights	Crítica	 Implementado
TC041	Create Reservation With Different Payment Methods	Média	 Implementado

6.2.4 Testes de Sessions (API)


ID Teste	Descrição	Prioridade	Status
TC042	Get All Data Sessions With Public Access	Alta	✓ Implementado
TC043	Get Sessions With Pagination	Média	✓ Implementado
TC044	Get Sessions Filtered By Movie	Alta	✓ Implementado
TC045	Get Sessions Filtered By Theater	Alta	✓ Implementado
TC046	Get Sessions Filtered By Date	Alta	✓ Implementado
TC047	Get Session By Valid ID	Alta	✓ Implementado
TC048	Get Session By Invalid ID	Média	✓ Implementado
TC049	Get Non-Existent Session	Média	✓ Implementado
TC050	Create Data Session As Admin	Crítica	✓ Implementado
TC051	Create Data Session With Invalid Movie ID	Alta	✓ Implementado
TC052	Create Data Session With Invalid Theater ID	Alta	✓ Implementado

TC053	Create Data Session With Invalid Data	Alta	 Implementado
TC054	Update Data Session As Admin	Crítica	 Implementado
TC055	Update Data Session With Partial Data	Alta	 Implementado
TC056	Update Non-Existent Session	Média	 Implementado
TC057	Delete Data Session As Admin	Alta	 Implementado
TC058	Delete Non-Existent Session	Média	 Implementado
TC059	Create Data Session Without Admin Rights	Crítica	 Implementado
TC060	Update Data Session Without Admin Rights	Crítica	 Implementado
TC061	Delete Data Session Without Admin Rights	Crítica	 Implementado
TC062	Reset Data Session Seats As Admin	Média	 Implementado
TC063	Reset Seats Without Admin Rights	Alta	 Implementado







TC064	Reset Seats Of Non-Existent Session	Baixa	 Implementado
TC065	Complete Session CRUD Flow	Crítica	 Implementado

6.2.5. Testes de Setup (API)

ID Teste	Descrição	Prioridade	Status
TC066	Create Admin User Successfully	Crítica	 Implementado
TC067	Create Admin User With Duplicate Email	Alta	 Implementado
TC068	Create Admin User With Invalid Data	Alta	 Implementado
TC069	Create Admin User With Weak Password	Alta	 Implementado
TC070	Create Test Users Successfully	Crítica	 Implementado
TC071	Create Test Users Multiple Times	Média	 Implementado
TC072	Test Users Should Include Admin And Regular User	Alta	 Implementado

TC073	Complete Setup Flow	Crítica	 Implementado
TC074	Setup Endpoints Response Time	Baixa	 Implementado

6.2.6. Teste das Theaters (API)

ID Teste	Descrição	Prioridade	Status
TC075	Get All Theaters Successfully	Alta	 Implementado
TC076	Get Theaters With Type Filter	Média	 Implementado
TC077	Get Theaters With Pagination	Média	 Implementado
TC078	Get Theater By Valid ID	Alta	 Implementado
TC079	Get Theater By Invalid ID	Média	 Implementado
TC080	Create Theater As Admin Successfully	Crítica	 Implementado
TC081	Create Theater Without Authentication	Crítica	 Implementado
TC082	Create Theater With Invalid Data	Alta	 Implementado
TC083	Create Theater With Duplicate Name	Alta	 Implementado
TC084	Update Theater Successfully	Crítica	 Implementado

TC085	Update Theater Without Authentication	Crítica	✓ Implementado
TC086	Update Non-Existent Theater	Média	✓ Implementado
TC087	Delete Theater Successfully	Alta	✓ Implementado
TC088	Delete Theater Without Authentication	Crítica	✓ Implementado
TC089	Delete Non-Existent Theater	Média	✓ Implementado
TC090	Theater CRUD Flow	Crítica	✓ Implementado
TC091	Create Multiple Theaters With Different Types	Média	✓ Implementado
TC092	Theater Capacity Validation	Baixa	✓ Implementado

6.2.7. Resumo por módulos de Teste.

Módulo	Total	Implementados	Percentual
Autenticação	6	6	100%
Filmes	22	22	100%
Reservas	13	13	100%
Sessões	24	24	100%
Setup	9	9	100%
Salas/Teatros	18	18	100%

TOTAL	92	92	100%
--------------	-----------	-----------	-------------

6.2.8. Tabela de criticalidade.

Prioridade	Quantidade	Percentual
Crítica	31	33.7%
Alta	42	45.7%
Média	17	18.5%
Baixa	2	2.2%

7. Cobertura e Resultados

Tipo de Teste	Cobertura Estimada	Observações
API	80%	Todos os endpoints principais validados
UI	70%	Fluxos críticos de navegação e reserva cobertos
E2E	100%	Cenário completo de ponta a ponta testado
Testes Negativos	100%	Principais cenários de erro contemplados
Rotas Admin	0%	Não aplicável (instabilidade no backend e frontend)

Resultado geral: todos os testes implementados executam com sucesso e validam as principais funções de negócio.

8. Limitações Identificadas

- As rotas de administração não estão estáveis, inviabilizando testes relacionados.

9. Conclusão

A automação de testes do sistema Cinema apresenta uma estrutura madura, modular e de fácil manutenção.

O conjunto de testes cobre adequadamente os fluxos principais e críticos do sistema, incluindo autenticação, listagem de filmes, sessões e fluxo de reserva.

O projeto demonstra:

- Adoção de boas práticas de QA e automação;
- Validação técnica de integrações entre frontend e backend;
- Alta taxa de sucesso na execução automatizada dos testes.

Embora existam oportunidades de expansão, a automação atual atende plenamente aos objetivos propostos, servindo como base sólida para evolução contínua.

10. Considerações Finais

O projeto de automação entregue está tecnicamente consistente, bem estruturado e funcional.

Ele garante que as principais jornadas do usuário e integrações de sistema estejam validadas automaticamente, contribuindo para a qualidade contínua do produto.

Com a adição futura de pipelines e testes complementares de performance e filtros, o projeto atingirá um nível de excelência corporativa em automação de QA.