

Relatório de Execução de Testes – Cinema App API

Data de Execução: 26 de junho de 2025 **Ferramenta de Automação:** Robot Framework

1. Resumo da Execução [↗](#)

Os testes automatizados da API REST foram realizados com sucesso utilizando o **Robot Framework**. As principais bibliotecas empregadas foram:

- **RequestsLibrary:** para a execução de requisições HTTP (GET, POST, PUT, DELETE).
- **JSONLibrary:** para a extração de dados e validação de respostas em formato JSON.
- **Collections:** para a manipulação de estruturas de dados, como dicionários para a montagem de *request bodies* e *headers*.
- **DateTime:** para a geração de dados dinâmicos (timestamps), garantindo a unicidade em cenários de criação.

A estrutura de automação seguiu rigorosamente o modelo proposto em `estrutura.png`, com uma organização modular que separa casos de teste, keywords de negócio, recursos e variáveis, garantindo alta legibilidade e manutenibilidade.

2. Limitações na Documentação [↗](#)

Durante a fase de implementação dos testes, foi identificado que a documentação da API via **Swagger** estava **incompleta** ou **inexistente** para as seguintes rotas críticas:

- **/sessions**
- **/reservations**

Essas rotas não apresentavam uma definição clara dos schemas de *request body* para as operações de `POST`, nem exemplos de uso. Essa lacuna impossibilitou a consulta direta à documentação pública como fonte primária para a criação dos testes destas funcionalidades.

3. Ação Técnica Realizada [↗](#)

Para contornar a limitação da documentação, foi adotada uma abordagem de **análise estática do código-fonte do backend** (fornecido no arquivo `.xml`). Essa análise permitiu compreender a fundo o comportamento esperado da API, incluindo:

- **Métodos e Rotas:** Verificação dos endpoints disponíveis em `src/routes/sessionRoutes.js` e `src/routes/reservationRoutes.js`.
- **Parâmetros e Corpo da Requisição:** Análise dos respectivos *controllers* (ex: `sessionController.js`) e *models* (ex: `Session.js`, `Reservation.js`) para identificar os campos obrigatórios, tipos de dados e as validações internas.

Com base neste levantamento, foi possível **automatizar os testes das rotas /sessions e /reservations com total precisão**, garantindo a cobertura planejada mesmo sem o apoio da documentação oficial.

4. Evidências e Logs [↗](#)

Todos os testes foram executados e os **logs completos foram gerados com o Robot Framework**. Os relatórios detalhados, que servem como evidência da execução e dos resultados, podem ser consultados nos seguintes arquivos gerados na pasta de resultados:

- `log.html` (Log detalhado com cada passo da execução)
- `report.html` (Relatório gerencial com gráficos e resumo dos resultados)

- `output.xml` (Arquivo de saída com todos os dados brutos da execução)

O ciclo de testes completo pode ser analisado através do **QAlity** afim de obter um histórico completo da execução de testes realizadas.

4.1 Issues Encontradas

Durante o processo de desenvolvimento da automação, foram identificados os seguintes pontos que exigiram análise e correção. Eles foram registrados na ferramenta de gestão de projetos para garantir o rastreamento e a resolução formal pela equipe de desenvolvimento.

1. Documentação da API Incompleta no Swagger

- **Descrição:** As rotas de `/sessions` e `/reservations` não possuíam documentação adequada no Swagger. Faltavam schemas de `request body` e exemplos, o que impedia o uso das rotas com base apenas na documentação pública.
- **Ação Tomada:** A limitação foi contornada através da análise direta do código-fonte do backend, especificamente os `controllers` e `models` da API, para mapear os dados necessários para as requisições. A automação foi desenvolvida com base nessa análise.
- **Links para o Jira:**
 - [BC262 - SESSIONS - Documentação Swagger Incompleta Impossibilita Testes](#)
 - [BC263 - RESERVATIONS - Documentação Swagger Incompleta Impossibilita Testes](#)

2. Inconsistência nas Credenciais de Teste do Administrador

- **Descrição:** Foi identificada uma divergência entre as senhas do usuário administrador (`admin@example.com`) nos diferentes arquivos do projeto (documentação vs. scripts de seed/validação), o que causou uma falha inicial de `401 Unauthorized` na suíte de testes.
- **Ação Tomada:** O ambiente de teste foi estabilizado utilizando as credenciais corretas no arquivo `variables.robot` , permitindo que o setup da suíte (login do admin) fosse executado com sucesso e os testes pudessem prosseguir.
- **Link para o Jira:** [BC264 - Inconsistência nas Credenciais de Teste do Administrador](#)

5. Arquivo de Apoio – Excel

Para facilitar a consulta e o gerenciamento dos resultados em outras ferramentas, foi gerado um arquivo de apoio em formato Excel:

- [Text Execution Report.xlsx](#)

Este arquivo contém uma transposição dos dados da tabela acima, com todos os cenários, status de execução, mensagens de erro (quando aplicável em depuração) e referências para as evidências nos logs, simulando a integração com um sistema de planejamento como o QAlity.

6. Conclusão

A execução da suíte de testes automatizados foi concluída com **100% de sucesso**. Todos os cenários descritos no planejamento foram implementados, depurados e validados. A ausência de documentação em rotas importantes foi um desafio superado com sucesso através da análise técnica do código-fonte.

O projeto de automação resultante é **completo, robusto, confiável e pronto para ser utilizado em ciclos de regressão**, garantindo a qualidade contínua da Cinema App API.