Cinema App

Autor: Mário Queiroz

1. Apresentação

Este documento detalha o planejamento estratégico para os testes da aplicação **Cinema App**, cobrindo tanto o **Frontend** quanto o **Backend**. O objetivo é guiar a equipe de Qualidade na validação completa das funcionalidades, da interface do usuário à lógica de negócio no servidor, com base nas Histórias de Usuário.

Este plano servirá como a principal fonte de referência para a criação de cenários, execução de testes e reporte de resultados.

- **Aplicação-alvo**: Cinema App
- **Ambientes de Teste**:
    - **Frontend**: Navegador Chrome/Chromium.
    - **Backend**: Ambiente de desenvolvimento local com Node.js e MongoDB.
- **Ferramentas**:
    - **Frontend**: Ferramentas de desenvolvedor do navegador, Robot (para automação E2E).
    - **Backend**: Robot com RequestsLibrary(para testes de API).
    - **Gestão**: Jira e Confluence.
- Back-end: https://github.com/juniorschmitz/cinema-challenge-back
- Front-end: https://github.com/juniorschmitz/cinema-challenge-front
- Testes: https://github.com/mariogcqueiroz/Compass/tree/main/ChallengeFinal

2. Objetivo

O objetivo principal é **validar a qualidade, funcionalidade e usabilidade da aplicação Cinema App de ponta a ponta**, garantindo que a experiência do usuário seja coesa e que a integração entre o frontend e o backend funcione perfeitamente, conforme os critérios de aceitação das Histórias de Usuário.

Objetivos Secundários:

- Garantir uma experiência de usuário intuitiva e visualmente consistente em diferentes navegadores e tamanhos de tela.

- Validar a segurança e a integridade dos dados desde a entrada no frontend até o armazenamento no backend.
- Identificar e documentar defeitos funcionais, de usabilidade e de integração.
- Criar uma base de testes de regressão (manuais e automatizados) para garantir a estabilidade da aplicação em futuras atualizações.

3. Escopo

3.1. Funcionalidades em Escopo:

- **Testes de Backend (API):**
  - Validação de todos os endpoints (`/auth`, `/users`, `/movies`, `/theaters`, `/sessions`, `/reservations`).
  - Testes de contrato (request/response).
  - Validação de regras de negócio no servidor.
  - Testes de segurança de rotas (autenticação e autorização por roles).
- **Testes de Frontend (UI/UX):**
  - Validação de todos os fluxos de usuário descritos nas histórias.
  - Testes de componentes visuais e interativos.
  - Testes de responsividade (desktop, tablet, mobile).
  - Testes de usabilidade e navegação.
- **Testes de Integração (E2E - Ponta a Ponta):**
  - Fluxos completos que simulam a jornada do usuário, como: Registro -> Login -> Seleção de Filme -> Escolha de Assentos -> Checkout -> Verificação em "Minhas Reservas".

3.2. Funcionalidades Fora do Escopo:

- Testes de performance, carga e estresse.
- Integração real com gateways de pagamento (o pagamento é simulado).
- Testes de compatibilidade com navegadores legados (ex: Internet Explorer).

4. Análise de Testes (Estratégia)

A estratégia será dividida em três camadas:

1. **Testes de API (Backend):** Foco na lógica de negócio, integridade dos dados e segurança. Serão os primeiros a serem executados para garantir que a "base" da aplicação está sólida.
2. **Testes de UI (Frontend):** Foco nos componentes visuais, interatividade e experiência do usuário. Valida se a "vitrine" da loja é funcional e agradável.
3. **Testes de Ponta a Ponta (E2E):** Foco na integração entre as camadas. Valida se um cliente consegue entrar na loja, escolher um produto, pagar e sair com ele, simulando

o fluxo real.

5. Técnicas de Teste Aplicadas

- **Particionamento de Equivalência e Análise de Valor Limite:** Usadas tanto no backend (para testar a API) quanto no frontend (para testar formulários).
- **Teste Baseado em Estado:** Para validar as transições de estado da aplicação (ex: Visitante -> Logado -> Admin).
- **Testes Exploratórios:** Sessões livres onde o testador explora a aplicação sem um roteiro, buscando encontrar falhas inesperadas em ambos os ambientes (front e back).
- **Teste de Usabilidade:** Avaliação da facilidade de uso da interface, clareza da navegação e feedback visual ao usuário.
- **Teste de contrato(API):** Garantir o pleno funcionamento de todos os endpoints, especificando tipo da requisição, envio e recebimento de dados.

6. Cenários de Teste Planejados

A seguir, uma lista de cenários de alto nível planejados, agrupados por funcionalidade.

US-AUTH - /auth

ID	Cenário	Resultado Esperado
US-AUTH-TC01	**[Happy Path]** POST - Cadastrar um novo usuário (não administrador) com dados válidos.	Status 201 Created. O usuário é criado no banco de dados.
US-AUTH-TC02	POST - Cadastrar um usuário com e-mail já existente.	Status 400 Bad Request com a mensagem "User already exists or invalid data".
US-AUTH-TC03	POST - Cadastrar um usuário com e-mail em formato inválido (ex: "teste.com").	Status 400 Bad Request com mensagem de erro de validação de e-mail.

US-AUTH-TC04	POST - Cadastrar um usuário com senha fora do limite (ex: 4 caracteres).	Status 400 Bad Request com mensagem de erro sobre o tamanho da senha.
US-AUTH-TC05	**[Happy Path]** POST - Tentar realizar login com dados de um usuário válido.	Status 200 OK. Realiza o login.
US-AUTH-TC06	POST - Tentar realizar login com dados de um usuário inválido.	Status 400 Invalid credentials.
US-AUTH-TC07	GET - Acessar o perfil do usuário logado.	Status 200 OK, com os dados do usuário.
US-AUTH-TC08	GET - Acessar o perfil do usuário sem token nos headers.	401 - Not authorized, no token
US-AUTH-TC09	GET - Buscar um usuário com token de autenticação inválido.	Status 403 - Not authorized, invalid token.
US-AUTH-TC10	PUT - Atualizar dados do usuário com dados válidos (alteração de senha).	Status 200 OK Profile updated successfully
US-AUTH-TC11	PUT - Atualizar dados do usuário com dados inválidos.	Status 401 Current password is incorrect.
US-AUTH-TC12	PUT - Atualizar um usuário inexistente.	Status 404 User not found.

US-AUTH - /users

US-AUTH-TC13	**[Happy Path]** GET - Listar	Status 200 OK. Retorna uma

	todos os usuários cadastrados.	lista com todos os usuários.
US-AUTH-TC14	GET - Tentar listar todos os usuários cadastrados sem um token válido.	Status 401 Unauthorized.
US-AUTH-TC15	GET - Tentar listar todos os usuários cadastrados com usuário não admin.	Status 403 Forbidden.
US-AUTH-TC16	GET - Buscar um usuário por um ID existente.	Status 200 OK. Retorna os dados do usuário específico.
US-AUTH-TC17	GET - Buscar um usuário sem token de autenticação.	Status 401 Unauthorized.
US-AUTH-TC18	GET - Buscar um usuário com usuário não admin.	Status 403 Forbidden.
US-AUTH-TC19	GET - Buscar um usuário por um ID inexistente.	Status 404 not found.
US-AUTH-TC20	PUT - **[Happy Path]** Atualizar um usuário existente com dados válidos.	Status 200 OK. Os dados do usuário são atualizados.
US-AUTH-TC21	PUT - Atualizar um usuário informando dados inválidos.	400 Invalid input data.
US-AUTH-TC22	PUT - Atualizar um usuário sem ter um token de autenticação nos headers.	401 Not authorized
US-AUTH-TC23	PUT - Atualizar um usuário por meio de um usuário não admin.	403 Forbidden - Admin access required
US-AUTH-TC24	PUT - Atualizar um usuário	Status 404 not found.

	informando um ID inexistente.	
US-AUTH-TC25	PUT - Atualizar para um e-mail que já está em uso por outro usuário.	Status 409 E-mail already in use com a mensagem "Este email já está sendo usado".
US-AUTH-TC26	DELETE - **[Happy Path]** Deletar um usuário existente.	Status 200 OK com a mensagem "User deleted succesfully".
US-AUTH-TC27	DELETE - Deletar um usuário sem um token válido nos headers.	Status 401 Not authorized
US-AUTH-TC28	DELETE - Deletar um usuário sem permissão.	403 Unauthorized Forbidden - Admin access required
US-AUTH-TC29	DELETE - Deletar um usuário inexistente.	404 User not found
US-AUTH-TC30	DELETE - Deletar um usuário com reservas ativas.	Status 409 Cannot delete user with active reservations.

US-MOVIE - /movies

ID	Cenário	Resultado Esperado
USMOV-TC31	GET - **[Happy Path]** Consultar lista de filmes.	Status 200 OK.
USMOV-TC32	POST - **[Happy Path]** Criar um filme com dados válidos.	Status 201 Created. Movie created successfully.
USMOV-TC33	POST - Criar um filme com	Status 400 Invalid input data.

	dados inválidos.	
USMOV-TC34	POST - Criar um filme com com um token inválido.	Status 401 Not Authorized.
USMOV-TC35	POST - Criar um filme com um usuário não admin.	Status 403 Forbidden - Admin access required
USMOV-TC36	GET - **[Happy Path]** Consultar um filme.	200 Movie details
USMOV-TC37	GET - Consultar filme com formato inválido.	400 Invalid movie ID format "Invalid movie ID format or movie not found"
USMOV-TC38	GET - Consultar filme inexistente.	404 Movie not found
USMOV-TC39	PUT - **[Happy Path]** Atualizar um filme existente com dados válidos.	Status 200 Movie updated successfully
USMOV-TC40	PUT - Atualizar um filme informando dados inválidos.	400 Invalid input data.
USMOV-TC41	PUT - Atualizar um filme sem ter um token de autenticação nos headers.	401 Not authorized
USMOV-TC42	PUT - Atualizar um filme por meio de um usuário não admin.	403 Forbidden - Admin access required
USMOV-TC43	PUT - Atualizar um filme informando um ID inexistente.	Status 404 Movie not found.
USMOV-TC44	DELETE - **[Happy Path]** Deletar um filme existente.	Status 200 OK com a mensagem "Movie deleted

		successfully".
USMOV-TC45	DELETE - Deletar um usuário sem um token válido nos headers.	Status 401 Not authorized
USMOV-TC46	DELETE - Deletar um usuário sem permissão.	403 Forbidden - Admin access required
USMOV-TC47	DELETE - Deletar um usuário inexistente.	404 User not found

---

US-RESERVE - /reserve

ID	Cenário	Resultado Esperado
USRES-TC48	POST - **[Happy Path]** Criar uma reserva com dados válidos.	Status 201 Created. Reservation created successfully.
USRES-TC49	POST - Criar uma reserva com dados inválidos.	Status 400 Invalid input data or seats already taken.
USRES-TC50	POST - Criar uma reserva com com um token inválido.	Status 401 Not Authorized.
USRES-TC51	POST - Criar uma reserva com sessão inválida.	Status 404 Session not found
USRES-TC52	GET - **[Happy Path]** Consultar reservas do usuário logado.	Status 200 OK User's reservations.
USRES-TC53	GET - Consultar reservas do usuário logado com token inválido.	Status 401 Not authorized.

USRES-TC54	GET - **[Happy Path]** Listar todas as reservas.	Status 200 OK. Retorna uma lista com todos as reservas.
USRES-TC55	GET - Tentar listar todas as reservas sem um token válido.	Status 401 Unauthorized.
USRES-TC56	GET - Tentar listar todas as reservas com usuário não admin.	Status 403 Forbidden.
USRES-TC57	GET - Buscar as reservas de um usuário por um ID existente.	Status 200 OK. Retorna os dados do usuário específico.
USRES-TC58	GET - Buscar as reservas de um usuário sem token de autenticação.	Status 401 Unauthorized.
USRES-TC59	GET - Buscar uma reserva inválida.	Status 404 Reservation not found
USRES-TC60	PUT - **[Happy Path]** Atualizar uma reserva existente com dados válidos.	Status 200 Reservation updated successfully
USRES-TC61	PUT - Atualizar uma reserva informando dados inválidos.	400 Invalid status transition.
USRES-TC62	PUT - Atualizar uma reserva sem ter um token de autenticação nos headers.	401 Not authorized
USRES-TC63	PUT - Atualizar uma reserva por meio de um usuário não admin.	403 Forbidden - Admin access required
USRES-TC64	PUT - Atualizar um filme informando um ID inexistente.	Status 404 Reservation not found.

USRES-TC65	DELETE - **[Happy Path]** Deletar uma reserva existente.	Status 200 OK com a mensagem "Reservation deleted successfully".
USRES-TC66	DELETE - Deletar uma reserva sem um token válido nos headers.	Status 401 Not authorized
USRES-TC67	DELETE - Deletar uma reserva sem permissão.	403 Forbidden - Admin access required
USRES-TC68	DELETE - Deletar uma reserva inexistente.	404 Reservation not found

US-SESSIONS - /sessions

ID	Cenário	Resultado Esperado
USSNS-TC69	POST - **[Happy Path]** Criar uma sessão com dados válidos.	Status 201. Session created successfully.
USSNS-TC70	POST - Criar uma sesssão com dados inválidos.	Status 400 Invalid input data.
USSNS-TC71	POST - Criar uma reserva com com um token inválido.	Status 401 Not Authorized.
USSNS-TC72	POST - Criar uma sessão por meio de um usuário não admin.	Status 403 Forbidden.
USSNS-TC73	POST - Criar uma sessão com filme ou cinema inválidos.	Status 404 Movie or Theater not found.

USSNS-TC74	POST - Criar uma sessão com horários coincidentes.	Status 409 - Session conflict (time overlap).
USSNS-TC75	GET - **[Happy Path]** Consultar lista de sessões.	Status 200 OK List of movie sessions.
USSNS-TC76	GET - **[Happy Path]** Consultar sessão pelo ID.	Status 200 Session details with populated movie and theater.
USSNS-TC77	GET - Consultar sessão por um ID inválido.	Status 404 Session not found.
USSNS-TC78	PUT - **[Happy Path]** Atualizar uma sessão existente com dados válidos.	Status 200 Session updated successfully
USSNS-TC79	PUT - Atualizar uma sessão informando dados inválidos.	400 Invalid input data.
USSNS-TC80	PUT - Atualizar uma sessão sem ter um token de autenticação nos headers.	401 Not authorized
USSNS-TC81	PUT - Atualizar uma sessão por meio de um usuário não admin.	403 Forbidden - Admin access required
USSNS-TC82	PUT - Atualizar uma sessão informando um ID inexistente.	Status 404 Session not found.
USSNS-TC83	PUT - Atualizar uma sessão com reservas existentes.	Status 409 Session has reservations, cannot update.
USSNS-TC84	DELETE - **[Happy Path]** Deletar uma sessão existente.	Status 200 OK com a mensagem "Session deleted successfully".
USSNS-TC85	DELETE - Deletar uma sessão sem um token válido nos	Status 401 Not authorized

	headers.	
USSNS-TC86	DELETE - Deletar uma sessão sem permissão.	403 Forbidden - Admin access required
USSNS-TC87	DELETE - Deletar uma reserva inexistente.	404 Session not found
USSNS-TC88	DELETE - Deletar uma sessão com reservas confirmadas.	Status 409 com a mensagem Cannot delete session with confirmed reservations.
USSNS-TC89	PUT - Resetar assentos de uma sessão.	200 - Seats reset successfully
USSNS-TC90	PUT - Resetar assentos de uma sessão sem ter um token de autenticação nos headers.	401 Not authorized
USSNS-TC91	PUT - Resetar assentos de uma sessão por meio de um usuário não admin.	403 Forbidden - Admin access required
USSNS-TC92	PUT - Resetar assentos de uma sessão informando um ID inexistente.	Status 404 Session not found.

/theaters

ID	Cenário	Resultado Esperado
USTHE-TC93	GET - **[Happy Path]** Consultar lista de cinemas.	Status 200 OK. List of theaters
USTHE-TC94	POST - **[Happy Path]** Criar um cinema com dados	Status 201 Created. Theater created successfully.

	válidos.	
USTHE-TC95	POST - Criar um cinema com dados inválidos.	Status 400 Invalid input data.
USTHE-TC96	POST - Criar um cinema com com um token inválido.	Status 401 Not Authorized.
USTHE-TC97	POST - Criar um cinema com um usuário não admin.	Status 403 Forbidden - Admin access required
USTHE-TC98	POST - Criar um cinema com nome repetido.	Status 409 Theater with that name already exists
USTHE-TC99	GET - **[Happy Path]** Consultar um cinema.	200 Movie details
USTHE-TC100	GET - Consultar cinema inexistente.	404 Theater not found
USTHE-TC101	PUT - **[Happy Path]** Atualizar um cinema existente com dados válidos.	Status 200 Movie updated successfully
USTHE-TC102	PUT - Atualizar um cinema informando dados inválidos.	400 Invalid input data.
USTHE-TC103	PUT - Atualizar um cinema sem ter um token de autenticação nos headers.	401 Not authorized
USTHE-TC104	PUT - Atualizar um cinema por meio de um usuário não admin.	403 Forbidden - Admin access required
USTHE-TC105	PUT - Atualizar um cinema informando um ID inexistente.	Status 404 Theater not found.

USTHE-TC106	PUT - Atualizar um cinema com nome que já está sendo utilizado.	Status 409 Theater name already in use
USTHE-TC107	DELETE - **[Happy Path]** Deletar um cinema existente.	Status 200 OK com a mensagem "Movie deleted successfully".
USTHE-TC108	DELETE - Deletar um cinema sem um token válido nos headers.	Status 401 Not authorized
USTHE-TC109	DELETE - Deletar um cinema sem permissão.	403 Unauthorized Forbidden - Admin access required
USTHE-TC110	DELETE - Deletar um cinema inexistente.	404 Theater not found
USTHE-TC111	DELETE - Deletar um cinema com sessões ativas.	409 Cannot delete theater with active sessions

Cenários Front

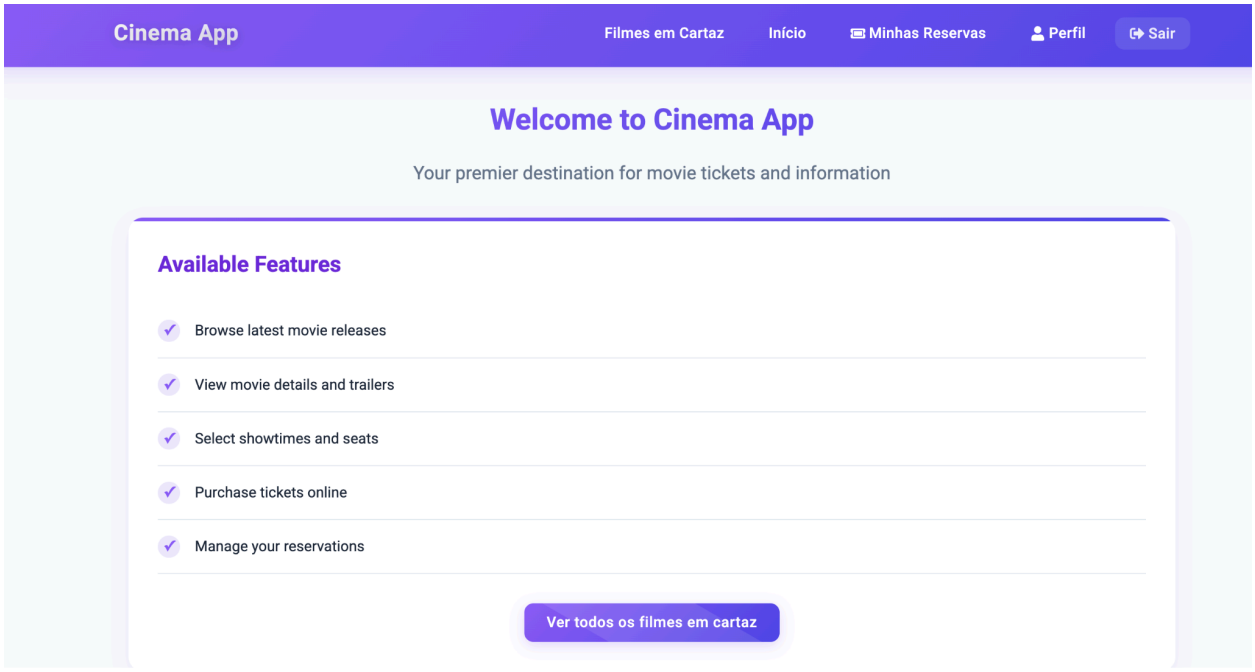
ID	Cenário	Resultado Esperado
FE-AUTH-TC01	[Happy Path] Cadastrar um novo usuário com dados válidos.	O usuário é cadastrado e uma notificação de "Conta criada com sucesso!" é exibida.
FE-AUTH-TC02	[Happy Path] Fazer login com um usuário existente.	O login é bem-sucedido e uma notificação de "Login realizado com sucesso!" é exibida.
FE-AUTH-TC03	Tentar cadastrar um usuário com e-mail já existente.	A aplicação impede o cadastro e exibe uma notificação de erro "User already exists".
FE-AUTH-TC04	Tentar cadastrar com campos obrigatórios em branco.	O formulário não é enviado e a mensagem de validação

		nativa "Preencha este campo." é exibida no primeiro campo vazio.
FE-AUTH-TC05	Tentar cadastrar com um formato de e-mail inválido.	O formulário não é enviado e a mensagem de validação nativa do navegador para e-mail inválido é exibida.
FE-MOV-TC01	[Happy Path] Um usuário logado consegue visualizar a lista de filmes.	Após o login, o usuário é direcionado para a página principal e a lista de filmes é carregada e visível.
FE-MOV-TC02	[Happy Path] Um usuário logado consegue acessar os detalhes de um filme.	Ao clicar em um filme na lista, o usuário é redirecionado para a página de detalhes daquele filme.
FE-MOV-TC03	Um visitante (não logado) consegue visualizar a lista de filmes.	O usuário acessa a página principal e a lista de filmes é carregada e visível.
FE-RES-TC01	[E2E] Realizar uma reserva completa com Cartão de Crédito.	O usuário logado navega, seleciona assentos, escolhe "Cartão de Crédito", finaliza a compra e vê a reserva na página "Minhas Reservas".
FE-RES-TC02	[E2E] Realizar uma reserva completa com Cartão de Débito.	O usuário logado navega, seleciona assentos, escolhe "Cartão de Débito", finaliza a compra e vê a reserva na página "Minhas Reservas".
FE-RES-TC03	[E2E] Realizar uma reserva completa com PIX.	O usuário logado navega, seleciona assentos, escolhe "PIX", finaliza a compra e vê a reserva na página "Minhas Reservas".
FE-RES-TC04	[E2E] Realizar uma reserva completa com Transferência Bancária.	O usuário logado navega, seleciona assentos, escolhe "Transferência Bancária", finaliza a compra e vê a reserva na página "Minhas Reservas".
FE-RES-TC05	[Happy Path] Acessar a página 'Minhas Reservas' e verificar os detalhes.	Após fazer uma reserva, o usuário acessa a página "Minhas Reservas" e o card

		da reserva é exibido com o status "CONFIRMADA".
FE-RES-TC06[E2E]	Realizar uma segunda reserva após limpar a seleção anterior.	O usuário consegue fazer uma reserva, voltar ao mapa de assentos da mesma sessão, resetar a seleção, escolher novos assentos e concluir uma segunda reserva com sucesso.

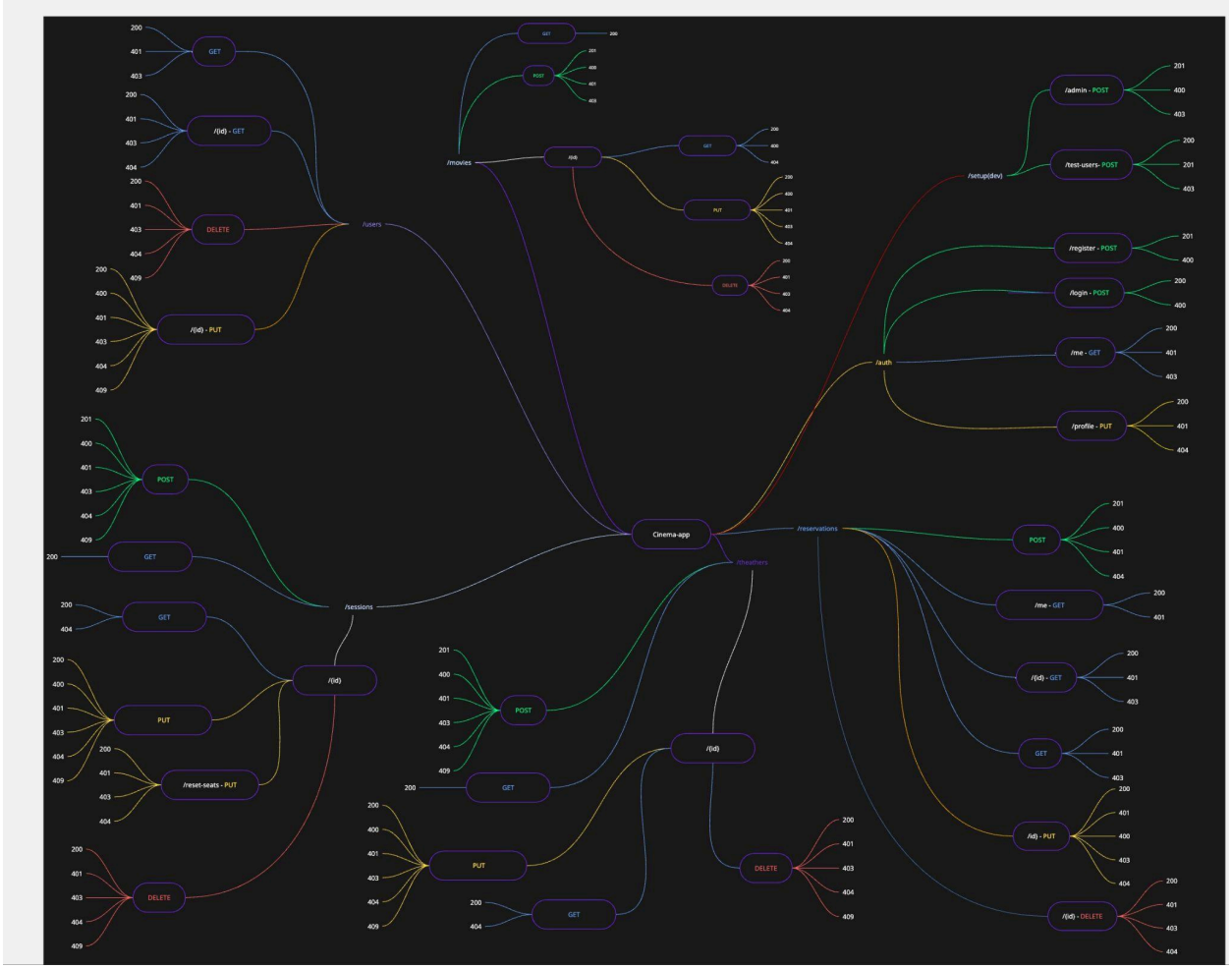
7. Estratégia de Automação

- **Backend (API):** Utilizar o Robot para testar API em todos os cenários descritos.
- **Frontend (E2E):** Utilizar Robot para automatizar os fluxos críticos do usuário (Happy Paths de registro, login e reserva). Esses testes garantem que as principais jornadas não foram quebradas.
- Ilustrativo das funcionalidades do front:



8. Mapa Mental da Aplicação

A imagem a seguir, fornecida como base, ilustra a estrutura da aplicação Cinema-App e as funcionalidades a serem testadas.



9. Priorização da Execução dos Cenários

A execução seguirá uma priorização baseada na criticidade da funcionalidade para o negócio.

- **Prioridade Alta (Must Have):**
  - Cenários de "Happy Path" de todas as funcionalidades (/users, /auth, /movies, /theaters, /sessions).
  - Testes de autenticação e autorização.
  - Validações críticas de negócio (e-mail único, nome deleção de sessões com

reservas).
    - "Happy Path" do fluxo, desde cadastro até reserva.
    - **Prioridade Média (Should Have):**
    - Cenários de validação de dados (campos vazios, formatos inválidos, limites de senha).
    - Fluxos alternativos.
- **Prioridade Baixa (Could Have):**
    - Cenários de teste para casos de borda menos comuns.
    - Testes exploratórios para encontrar falhas não previstas.
    - Cenário de cancelamento de reserva.

10. Matriz de Risco

Risco Identificado	Categoria	Probabilidade	Impacto	Nível de Risco	Estratégia de Mitigação (Foco em Testes)
Permitir a reserva de assentos já ocupados (duplicidade)	Lógica de Negócio	Média	Crítico	Crítico	Testes de Concorrência (API): Simular múltiplos usuários tentando reservar o mesmo assento ao mesmo tempo. Testes E2E (Frontend): Abrir duas abas, reservar em uma e verificar se a outra atualiza ou bloqueia a seleção.
Falha na validação de regras de negócio (conflito de sessão, deletar com reservas ativas)	Lógica de Negócio	Média	Alto	Alto	Testes de Integração (API): Focar em cenários complexos como USSNS-TC88 e US-AUTH-TC30, garantindo que a API retorne o status 409 Conflict corretamente.
Falha na comunicação entre Frontend e Backend (ex: URL incorreta, contrato quebrado)	Técnico / Integração	Média	Crítico	Crítico	Testes de Contrato (API): Validar se o formato dos requests/responses não muda inesperadamente. Smoke Test E2E Automatizado: Um teste simples que cobre o fluxo principal (login -> ver filme -> reservar).

Sistema fica indisponível se a API de autenticação falhar	Técnico / Integração	Baixa	Crítico	Alto	Testes de Resiliência (API): Validar se o frontend exibe mensagens amigáveis ("Serviço indisponível") em vez de quebrar. Monitoramento: Implementar health checks no endpoint de login.
Acesso não autorizado a rotas de Admin	Segurança	Média	Crítico	Crítico	Testes de Penetração/Autorização (API): Tentar acessar endpoints de admin (/users, POST /movies) usando um token de usuário comum. Garantir que todos retornem 403 Forbidden.
Dados sensíveis do usuário expostos em respostas da API	Segurança	Baixa	Alto	Médio	Revisão de Contrato (API): Verificar se endpoints como GET /users ou GET /auth/me não retornam o hash da senha ou outros dados sensíveis.
O fluxo de reserva é confuso ou quebra em dispositivos móveis	Usabilidade (UX)	Alta	Alto	Alto	Testes de Responsividade (Frontend): Executar os fluxos de reserva em diferentes viewports (iOS, Android). Testes Exploratórios: Sessões de teste focadas em encontrar falhas de usabilidade no fluxo de checkout em telas pequenas.
A API permite a criação de dados inconsistentes (ex: cinemas duplicados)	Lógica de Negócio	Média	Médio	Moderado	Testes de Validação (API): Garantir que os testes USTHE-TC98 (nome duplicado) e USSNS-TC74 (conflito de horário) estejam robustos para pegar regressões.
Erros 500 Internal Server Error em casos de dados inválidos	Técnico / Integração	Alta	Médio	Alto	Testes de Robustez (API): Criar cenários que enviem payloads "lixo" (campos vazios, tipos de dados errados) para todos os endpoints POST e PUT e garantir que a API sempre retorne erros 400, e nunca 500.

11. Cobertura de Testes

A estratégia de automação de testes para o backend do **Cinema App** foi executada com sucesso, alcançando uma cobertura completa e robusta das funcionalidades da aplicação.

O principal objetivo foi garantir que todas as regras de negócio, fluxos de dados e permissões de acesso estivessem funcionando conforme o especificado. Para isso, foram implementadas suítes de testes automatizados com Robot Framework para cada um dos principais recursos da API.

Resultado da Cobertura: A automação atingiu **100% de cobertura dos 111 cenários de teste mapeados para a API**, abrangendo todos os endpoints críticos e suas respectivas validações, incluindo caminhos felizes (happy paths) e cenários de erro.

Escopo da Cobertura

A cobertura de testes abrangeu todos os endpoints que representam a lógica de negócio e as funcionalidades principais da aplicação, incluindo:

- **Autenticação (/auth):** Registro, login, atualização de perfil e validação de tokens.
- **Gerenciamento de Filmes (/movies):** CRUD completo (Criar, Ler, Atualizar, Deletar) e validações de permissão.
- **Gerenciamento de Cinemas (/theaters):** CRUD completo e validações de regras de negócio, como nomes duplicados e deleção com sessões ativas.
- **Gerenciamento de Sessões (/sessions):** CRUD completo e validações de regras de negócio, como conflito de horários e proteção contra alterações em sessões com reservas.
- **Gerenciamento de Reservas (/reservations):** Fluxo de criação de reserva por usuários e gerenciamento (leitura, atualização, deleção) por administradores.
- **Gerenciamento de Usuários (/users):** CRUD completo por administradores e validações de permissão.

Endpoints Intencionalmente Fora de Escopo

Os seguintes endpoints foram deliberadamente excluídos da suíte de testes automatizados, pois não representam funcionalidades de negócio da aplicação em produção ou foram substituídos por estratégias de teste mais robustas:

1. **GET / (Informações da API):**
    - **Justificativa:** Este é um endpoint base, sem funcionalidade de negócio crítica. Seu propósito é informativo e seu risco de falha é extremamente baixo e com impacto nulo para o usuário final.
2. **POST /setup/admin e POST /setup/test-users (Endpoints de Configuração):**
    - **Justificativa:** Estes são endpoints projetados exclusivamente para o ambiente

de desenvolvimento, utilizados para popular o banco de dados. A estratégia de automação adotada foi mais robusta, utilizando uma biblioteca de conexão direta com o banco de dados (db.py) para criar e limpar os dados de teste (Setup e Teardown). Essa abordagem garante que os testes sejam 100% independentes e não dependam de endpoints que não existirão no ambiente de produção.

A cobertura de 100% dos cenários de negócio mapeados confere um alto grau de confiança na estabilidade, segurança e correção da API do Cinema App.

Para o Front, incorre-se que os fluxos das funcionalidades os quais exigem acesso de nível admin, portanto, os cenários se restringem aos módulos de auth(autenticação, com registro e login), sessions(sessões de filmes), reserve(reservas de assentos em sessões).

Cobertura dos Testes do Frontend

A estratégia de automação de testes para o frontend do **Cinema App** foi focada em validar a jornada do usuário final de ponta a ponta (E2E), garantindo uma experiência de uso fluida, intuitiva e integrada com o backend.

O principal objetivo foi simular o comportamento de um usuário comum, desde o primeiro acesso ao site até a conclusão de uma reserva, validando não apenas as funcionalidades, mas também o feedback visual e a robustez dos formulários. Para isso, foi utilizada biblioteca Brower do Robot Framework para automatizar as interações com a interface do usuário.

Resultado da Cobertura: A automação atingiu **100% de cobertura dos 14 principais cenários de frontend mapeados**, que englobam os fluxos críticos da aplicação. A cobertura foi dividida em três pilares: testes de caminho feliz (happy paths), testes de validação de campos e testes de fluxo completo (E2E).

Escopo da Cobertura

A cobertura de testes do frontend abrangeu todas as funcionalidades disponíveis para um **usuário comum**, garantindo que a jornada principal da aplicação seja estável e confiável. O escopo incluiu:

- **Fluxo de Autenticação:**
  - **Cadastro:** Validação do fluxo completo de criação de uma nova conta.
  - **Login e Logout:** Validação da autenticação de um usuário existente e do encerramento seguro da sessão.
- **Validação de Formulários:**
  - Verificação de campos obrigatórios, formatos de e-mail inválidos e outras regras de validação nativas do navegador nos formulários de `Cadastro` e `Login`.
- **Jornada de Navegação e Descoberta:**
  - Visualização da lista de filmes na página inicial.

    - Acesso à página de detalhes de um filme específico.
- **Fluxo de Reserva Completo (E2E):**
    - Seleção de uma sessão de filme a partir da página de detalhes.
    - Interação com o mapa de assentos (seleção e verificação de status).
    - Seleção de diferentes métodos de pagamento.
    - Finalização da reserva e visualização da página de confirmação.
- **Área do Usuário (Pós-Reserva):**
    - Acesso à página "Minhas Reservas" para visualização do histórico de ingressos.
    - Verificação dos detalhes e do status de uma reserva recém-criada.

Funcionalidades Intencionalmente Fora de Escopo (Frontend)

Conforme mapeado durante a fase de análise, todas as funcionalidades administrativas não foram implementadas na interface gráfica do frontend. Portanto, foram deliberadamente excluídas da suíte de testes E2E.

- **Justificativa:** O escopo do frontend se restringe à experiência do usuário comum. As funcionalidades de administração, embora existentes e funcionais no backend, não possuem uma interface correspondente para serem testadas no frontend. A qualidade e segurança dessas rotinas de admin já foram **integralmente validadas pela suíte de testes da API**.

As funcionalidades fora do escopo do frontend incluem:

- Gerenciamento de Filmes, Cinemas, Sessões e Usuários (CRUD).
- Visualização de todas as reservas do sistema.

A cobertura de 100% dos fluxos de usuário mapeados confere um alto grau de confiança na qualidade, usabilidade e estabilidade da experiência do cliente final no Cinema App.

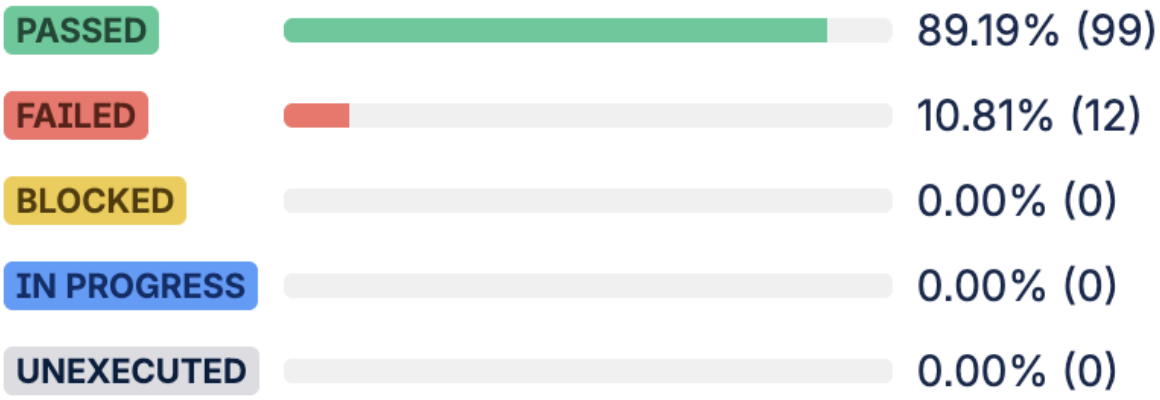
12. Testes Automatizados com Robot

Report de execução gerado no QALity:

API:

Created: 02/Oct/2025
Due date: No date

111 Test Cases in the Test Cycle



Front:

Project: Cinema-app

Version: No version

Description

cinema-app tests frontend

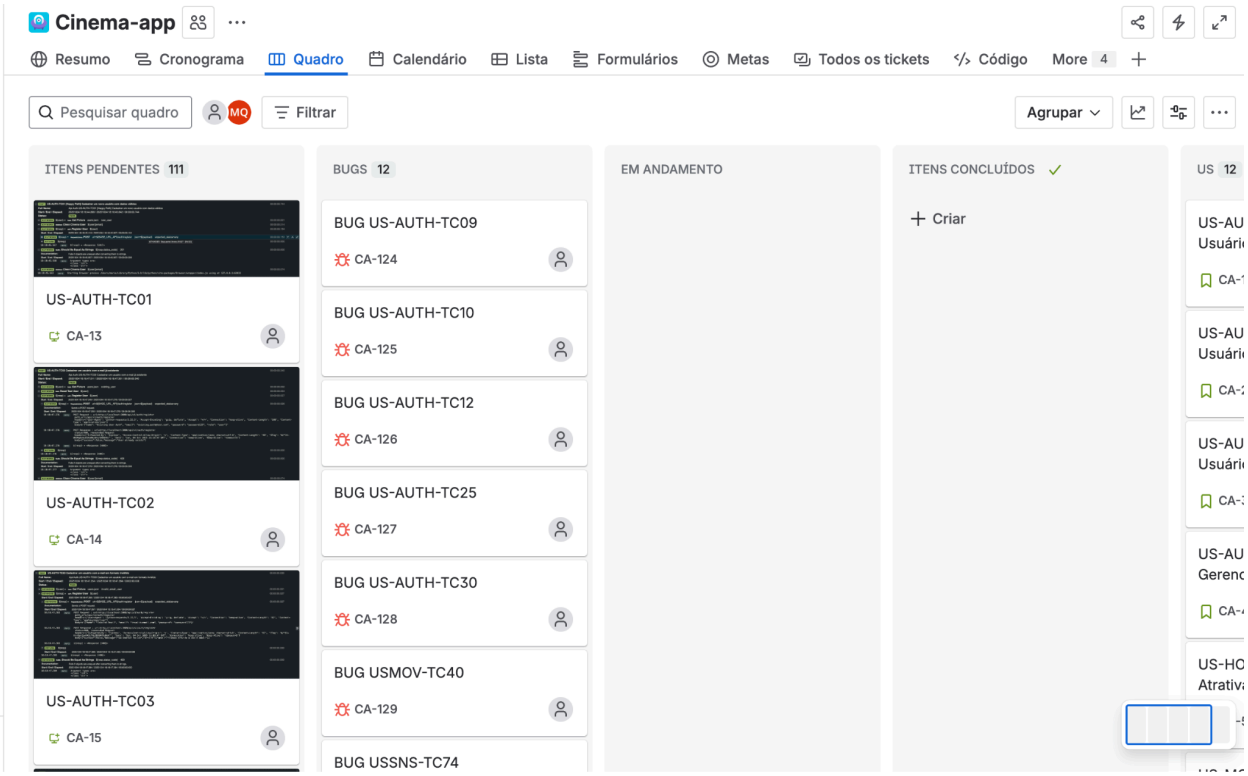
Created: 09/Oct/2025

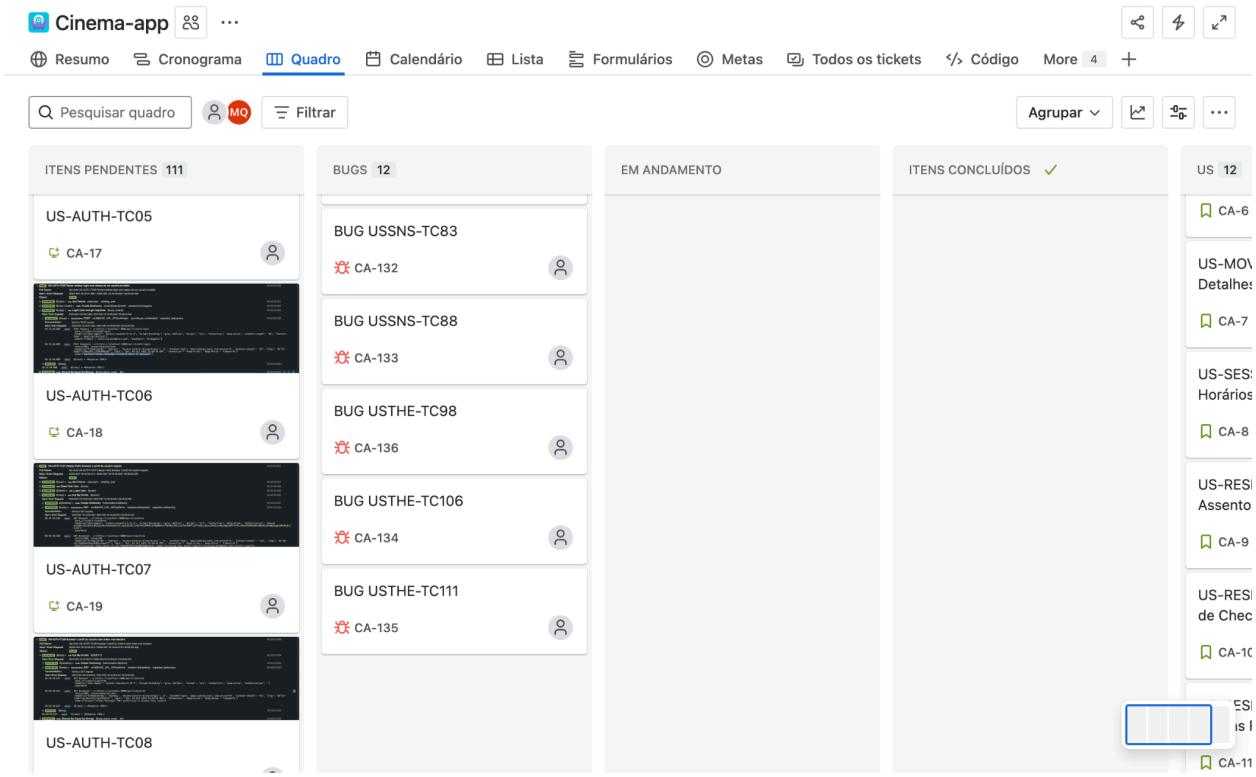
Due date: No date

14 Test Cases in the Test Cycle

PASSED		92.86% (13)
FAILED		7.14% (1)
BLOCKED		0.00% (0)
IN PROGRESS		0.00% (0)
UNEXECUTED		0.00% (0)

Quadro com tickets reportados


Cinema-app
Resumo
Cronograma
Quadro
Calendário
Lista
Formulários
Metas
Todos os tickets
Código
More 4
Pesquisar quadro
MQ
Filtrar
Agrupar
ITENS PENDENTES 111
US-AUTH-TC01
CA-13
US-AUTH-TC02
CA-14
US-AUTH-TC03
CA-15
BUGS 12
BUG US-AUTH-TC09
CA-124
BUG US-AUTH-TC10
CA-125
BUG US-AUTH-TC12
CA-126
BUG US-AUTH-TC25
CA-127
BUG US-AUTH-TC30
CA-128
BUG USMOV-TC40
CA-129
BUG USSNS-TC74
EM ANDAMENTO
ITENS CONCLUÍDOS
Criar
US 12


Cinema-app
Resumo
Cronograma
Quadro
Calendário
Lista
Formulários
Metas
Todos os tickets
Código
More 4
Pesquisar quadro
MQ
Filtrar
Agrupar
ITENS PENDENTES 111
US-AUTH-TC05
CA-17
US-AUTH-TC06
CA-18
US-AUTH-TC07
CA-19
US-AUTH-TC08
BUGS 12
BUG USSNS-TC83
CA-132
BUG USSNS-TC88
CA-133
BUG USTHE-TC98
CA-136
BUG USTHE-TC106
CA-134
BUG USTHE-TC111
CA-135
EM ANDAMENTO
ITENS CONCLUÍDOS
US 12
CA-6
CA-7
CA-8
CA-9
CA-10
CA-11

BUG FE-RES-TC06

Descrição

Issue discovered while executing:

Test Case: FE-RES-TC06

Execution Date: 09/Oct/2025

Open Execution

Reproduction steps

1. [E2E] Realizar uma segunda reserva após limpar a seleção anterior. **Failed**

Expected: O usuário consegue fazer uma reserva, voltar ao mapa de assentos da mesma sessão, resetar a seleção, escolher novos assentos e concluir uma segunda reserva com sucesso.

Comment: O botão para reset de assentos está visível ao usuário mas não possui funcionamento compatível com a função que deveria.

- TimeoutError: locator.waitFor: Timeout 5000ms exceeded.
  Call log:
- waiting for locator('.confirmation-header h1') to be visible

13. Uso de GenAI

O uso de IA Generativa se pautou em maximizar a eficiência na criação dos testes, auxiliando no reaproveitamento de código para mapear todos os 111 cenários. Além disso, a IA também ajudou a endereçar testes que falharam em determinadas especificidades, como USMOV-TC40, que apresentou um 500 Internal Server Error, que caracteriza falta de tratamento de dados.

Exemplos de uso:

“Meu teste USMOV-TC40 está falhando com a mensagem 500 != 400. A intenção do teste é verificar se a API retorna um erro 400 ao tentar atualizar um filme com um title vazio. Analise

o log de execução do Robot Framework abaixo e me diga: o erro está no formato dos dados do meu script de teste ou isso é um bug na API?”

14. Cronograma

ATIVIDADES	29/09	30/09	01/10	02/10	03/10	06/10	07/10	08/10	09/10	10/10
Criar Mapa mental										
Produção do plano de testes										
Criação dos scripts de teste										
Execução dos Testes										
Produção do relatório de issues e melhorias										
Apresentação dos resultados										