

# Hands-on Lab: Automação de Testes

## Sistema de Biblioteca (Versão CRE)

*Certificação Rumos Expert (CRE): Test Automation Engineer*

Automação de API

Automação de UI

Testes de Integração

### 1. Objetivos de Aprendizagem

Ao finalizar o exercício, o candidato deve demonstrar capacidade de:

- Automatizar casos de teste de **API** do backend (REST)
- Automatizar casos de teste de **interface** (frontend web)
- Validar **regras de negócio**, perfis de usuário (Aluno, Funcionário, Admin) e cenários de erro
- Estruturar o projeto de automação com **boa arquitetura**, reutilização e legibilidade
- Produzir **documentação clara** sobre execução e evidências dos testes

## 2. Contexto da Aplicação

O sistema de biblioteca inclui:

### 2.1 Autenticação

- Registro público (cria sempre **Aluno**)
- Login de usuários já cadastrados (Aluno, Funcionário, Admin)

### 2.2 Perfis e RBAC Simples

- **Aluno**: acesso a dashboard, livros, favoritos, arrendamentos, compras e "Minhas Compras"
- **Funcionário**: acesso a dashboard, livros, aprovação de arrendamentos, compras admin
- **Admin**: tudo acima + **Admin Usuários (CRUD)**

### 2.3 Gerenciamento de Livros (CRUD)

- Criar, listar, atualizar, remover livros
- Controle de **estoque** e **preço**
- Página de detalhes de cada livro

### 2.4 Funcionalidades Principais

- **Dashboard Dinâmico**: Estatísticas diferentes por tipo de usuário
- **Favoritos**: Adicionar, remover e listar favoritos
- **Arrendamentos**: Alunos solicitam; Funcionário/Admin aprovam ou rejeitam
- **Compras**: Alunos registram; Funcionário/Admin gerem (aprovar/cancelar)
- **Admin de Usuários**: CRUD completo de usuários

⚙️ **Acesso**: Todas as funcionalidades estão disponíveis em <http://localhost:3000> após `npm start`

### 3. Escopo dos Testes a Desenvolver

Os casos de teste detalhados estão separados em **dois documentos complementares**:

- **Documento 2:** Casos de Teste de **Backend (API)** – 32 casos
- **Documento 3:** Casos de Teste de **Frontend (UI)** – 25 casos

O candidato deve **automatizar todos os casos descritos** ou justificar claramente os que não forem implementados (por limitação de tempo, ferramenta, etc.).

### 4. Entregáveis

O candidato deve entregar:

1. **Código-fonte da automação** (API e UI) em um repositório público no GitHub
2. **README de testes** explicando:
  - Stack utilizada (linguagem, framework, libs)
  - Como instalar dependências
  - Como executar os testes de API e os de UI (comandos)
  - Como configurar dados de teste (se necessário)
3. **Relatório de execução**:
  - Gerado pela própria ferramenta (HTML report) e referenciado no README
  - Deve conter evidências de sucesso/falha (logs, screenshots em falhas, etc.)

## 5. Critérios de Avaliação (100%)

A nota final será calculada pela soma ponderada dos seguintes critérios:

Critério	Peso
<b>1. Organização da Estrutura do Projeto</b> <ul style="list-style-type: none"><li>Pastas separadas para testes de API e UI</li><li>Separação clara entre Page Objects, fixtures e configurações</li></ul>	<b>10%</b>
<b>2. Nomenclatura Adequada</b> <ul style="list-style-type: none"><li>Nomes claros e consistentes</li><li>Idioma único, padrão organizado</li></ul>	<b>10%</b>
<b>3. Utilização de Page Objects e Camadas</b> <ul style="list-style-type: none"><li>Uso de Page Objects para UI</li><li>Separação entre lógica e mapeamento</li></ul>	<b>12%</b>
<b>4. Cobertura Funcional dos Casos de Teste</b> <ul style="list-style-type: none"><li>Abrangência de cenários (API + UI)</li><li>Casos felizes e de erro/validação</li><li>Fluxos de cada perfil (Aluno, Funcionário, Admin)</li></ul>	<b>16%</b>
<b>5. Uso Adequado de Waits e Sincronização</b> <ul style="list-style-type: none"><li>Esperas explícitas (não sleeps fixos)</li><li>Testes estáveis, sem flakiness</li></ul>	<b>10%</b>
<b>6. Tratamento de Erros e Evidências</b> <ul style="list-style-type: none"><li>Manejo de exceções</li><li>Logs e screenshots automáticos em falhas</li></ul>	<b>8%</b>
<b>7. Manutenção e Reutilização de Código</b> <ul style="list-style-type: none"><li>Métodos utilitários para operações repetidas</li><li>Evitar duplicação excessiva</li></ul>	<b>12%</b>

8. Testes Independentes e Isolados <ul style="list-style-type: none"><li>• Execução independente sem ordem</li><li>• Setup/teardown ou dados dinâmicos</li></ul>	<b>10%</b>
9. Boas Práticas de Automação e Qualidade <ul style="list-style-type: none"><li>• Asserts claros com mensagens úteis</li><li>• Ausência de código morto</li></ul>	<b>6%</b>
10. Documentação e README de Testes <ul style="list-style-type: none"><li>• Instruções completas e testadas</li><li>• Explicação da arquitetura de testes</li></ul>	<b>6%</b>
<b>TOTAL</b>	<b>100%</b>

 **Para Obter a Certificação CRE:** Atingir **mínimo de 65%** na pontuação final.

## 6. Recomendações ao Candidato

- Comece pelos **cenários felizes de API** (registro, login, CRUD de livros) e depois amplie para erros e regras de negócio (estoque, perfis, etc.)
- Automatize primeiro **1 fluxo completo de UI** (ex.: login → dashboard → cadastrar livro → verificar na lista) antes de expandir
- Reutilize dados de teste e, quando possível, **crie dados via API** antes de executar os testes de UI
- Rode a suíte localmente várias vezes para garantir **estabilidade antes da submissão**
- Organize o código com **Page Objects** desde o início para facilitar manutenção
- Implemente **logs detalhados** e **screenshots em falhas** para auxiliar no debug

## 7. Próximos Documentos

### Documento 2: Casos de Teste – Backend (API)

Contém **32 casos de teste de API** distribuídos em 7 categorias:

1. **Autenticação e Perfis:** 4 casos
2. **Livros:** 8 casos
3. **Estatísticas:** 1 caso
4. **Favoritos:** 4 casos
5. **Arrendamentos:** 5 casos
6. **Compras:** 6 casos
7. **Admin Usuários:** 4 casos

Cada caso inclui: endpoint, objetivo, pré-condições, dados de entrada (JSON), e validações esperadas detalhadas.

### Documento 3: Casos de Teste – Frontend (UI)

Contém **25 casos de teste de UI** distribuídos em 9 categorias:

1. **Registro e Login:** 4 casos
2. **Proteção de Rotas e Navegação:** 3 casos
3. **Dashboard:** 2 casos
4. **Livros:** 3 casos
5. **Favoritos:** 3 casos
6. **Arrendamentos:** 2 casos
7. **Compras:** 2 casos
8. **Admin Usuários:** 4 casos
9. **Logout:** 1 caso

Cada caso inclui: objetivo, pré-condições (quando aplicável), passos detalhados, e validações esperadas.

 **Importante:** Este é o documento 1 de 3. Consulte os documentos 2 e 3 para os casos de teste detalhados (API e UI).