Documentação da Coleção de Testes de API - Fake Store API

API Utilizada: Fake Store API (simulação de e-commerce REST)
Objetivo: Validar o funcionamento dos métodos HTTP (GET, POST, PUT, DELETE) e garantir a integridade da resposta (Status Code, Body e Headers).

1. Cobertura de Métodos

A coleção **"Suite master"** cobre os seguintes métodos, conforme solicitado:

Método	Item de Teste (Exemplo)	Objetivo
GET	1 - Todos os produtos	Consulta e valida o schema da listagem completa.
POST	6 - Novo produto	Cria um novo recurso e valida o status `201 Created` e o retorno do `ID`.
PUT	11 - PUT Atualização Completa de Produto (A ser adicionado)	Substitui completamente um recurso existente e verifica os dados atualizados.
DELETE	9 - DELETE Produto	Remove um recurso e verifica o status `200 OK` e o objeto retornado.

2. Validações Implementadas

Todos os testes utilizam o framework `pm.test` do Postman para aplicar as seguintes verificações:

A. Validação de Status Code

Em cada requisição, é validado o código de status HTTP:

- `GET` (Sucesso): `pm.response.to.have.status(200);`
- `POST` (Criação): `pm.response.to.have.status(201);`
- `PUT` (Sucesso): `pm.response.to.have.status(200);`
- `GET` (Não Encontrado): `pm.response.to.have.status(404);`

B. Validação de Body (Corpo da Resposta)

São aplicadas validações de integridade e conteúdo:

1. **Validação de Schema:** Verifica a presença de chaves obrigatórias (ex: `'id', 'title', 'price'`).
2. **Validação de Tipo:** Garante que os campos numéricos (ex: `price`, `id`) são do tipo `number`.
3. **Validação de Conteúdo/Formato:** Verifica o comprimento de arrays (ex: `?limit=5` deve ter `.lengthOf(5)`) ou a inclusão de valores específicos.
4. **Validação de Propriedades Específicas:** Verifica se os dados enviados em `POST` ou `PUT` foram refletidos na resposta (ex: `pm.expect(jsonData.title).to.include(...)`).

C. Validação de Headers (Cabeçalhos)

O teste `3 - Produtos por categorias` verifica a presença e o valor do cabeçalho `Content-Type`:

- `pm.response.to.have.header("Content-Type", "application/json; charset=utf-8");`

3. Estrutura da Coleção (Pastas)

A coleção está organizada por níveis de complexidade, facilitando a execução de testes regressivos focados:

- **1 - Nivel Básico:** Foco em consultas simples (`GET`).
- **2 - Nível Intermediário:** Inclui testes de limitação (`limit`), cenários negativos (`404`), e o primeiro teste de modificação (`POST`).
- **3 - Nível Avançado:** Contém testes de modificação complexa (`PUT`, `DELETE`) e autenticação (`Login de Usuário`).
- **4 - Multiplos Query Params:** Testa a combinação de múltiplos parâmetros de URL (ex: `limit` e `sort`).