UNIVERSIDADE ESTADUAL DO TOCANTINS - UNITINS
ENGENHARIA DE QUALIDADE

FERRAMENTAS E EXEMPLOS PRÁTICOS DE TESTES DE SOFTWARE

Joana Letícia Azevedo do Nascimento Soares

Palmas - TO
2025

SUMÁRIO

1. INTRODUÇÃO

Um dos aspectos mais relevantes durante o desenvolvimento de sistemas é a qualidade de software, uma vez que ela impacta diretamente na confiabilidade, usabilidade e desempenho das aplicações. Nesse contexto, os testes de software desempenham um papel essencial ao permitir a verificação e validação do comportamento do sistema frente aos requisitos estabelecidos.

Os testes são os responsáveis por identificar falhas, validar funcionalidades e garantir que modificações no código não gerem novos erros. Além disso, contribuem para a manutenção da segurança, da performance e da experiência do usuário.

Dentre os diversos tipos de testes existentes, os que mais se destacam são os testes unitários, de integração, de sistema, de aceitação e de regressão. Cada tipo possui seu papel dentro do ciclo de vida do software, desde a validação de pequenas unidades de código até a verificação de requisitos funcionais completos.

Sendo assim, este relatório apresenta um estudo teórico e prático sobre testes de software aplicados a APIs REST utilizando a ferramenta Postman, uma das mais utilizadas atualmente no mercado para esse tipo de teste. A proposta será demonstrar, de forma prática, como criar requisições, automatizar verificações e validar respostas de uma API, simulando um cenário real de uso.

2. TIPOS DE TESTES ABORDADOS

Durante a elaboração do projeto prático com a ferramenta Postman, foram aplicados diferentes tipos de testes voltados à verificação do comportamento de uma API REST. Os testes foram criados com o objetivo de simular situações comuns no consumo de APIs, garantindo que as respostas estejam corretas e que os dados sejam manipulados adequadamente.

2.1. Testes de Requisição HTTP (Testes Funcionais de API)

Esse tipo de teste verifica se as requisições feitas à API retornam respostas válidas de acordo com os métodos HTTP utilizados (GET, POST, PUT, DELETE).

Por exemplo, ao realizar uma requisição `GET /posts`, espera-se que o servidor responda com o código 200 (OK) e retorne uma lista de objetos no formato JSON.

2.2. Testes de Integração Simples

Os testes de integração foram representados pela simulação do encadeamento entre diferentes endpoints da API. Por exemplo, após criar um novo post via `POST /posts`, foi verificado se esse novo recurso pode ser recuperado com `GET /posts/{id}`. Esse tipo de teste assegura que as partes do sistema se comunicam corretamente entre si.

2.3. Testes de Regressão (Simples)

Foram aplicadas requisições repetidas após alterações nos dados, para verificar se modificações em um recurso não impactam negativamente o comportamento da API. Por exemplo, após atualizar um post com `PUT`, foi feita uma nova requisição `GET` para confirmar a persistência dos dados alterados.

2.4. Testes de Aceitação (Básicos)

Os testes de aceitação simulam a experiência do usuário final ao consumir a API. Eles verificam se os principais requisitos funcionais estão sendo atendidos, como a presença de campos esperados nas respostas (`title`, `body`, `userId`), o tempo de resposta, e o código de status HTTP correto.

2.5. Validações Automatizadas com Scripts

O Postman permite a criação de scripts personalizados para automatizar as verificações. Nos testes realizados, foram utilizados scripts na aba **Tests** das requisições para validar, por exemplo, o código de status da resposta (200, 201, 204) e a existência de propriedades esperadas em objetos JSON retornados.

3. POSTMAN

3.1. Nome e Origem

O Postman é uma plataforma popular para desenvolvimento, teste e documentação de APIs (Application Programming Interfaces). Criado inicialmente como uma extensão para o Google Chrome em 2012, rapidamente ganhou popularidade e evoluiu para uma aplicação desktop independente, com versões para Windows, macOS e Linux.

Por ser uma solução amplamente utilizada por desenvolvedores, testers e equipes de DevOps, o Postman se destaca pela sua interface intuitiva, suporte a múltiplos protocolos e recursos avançados de automação.

3.2. Principais Recursos e Diferenciais

O Postman oferece um conjunto robusto de funcionalidades que o tornam ideal para testes de APIs RESTful. Entre os principais recursos utilizados neste projeto, destacam-se:

- Criação de coleções de requisições organizadas em pastas;
- Execução de testes automatizados com scripts em JavaScript;
- Validação de status, tempo de resposta e conteúdo JSON;
- Ambientes configuráveis com variáveis globais, locais e de ambiente;
- Geração automática de documentação interativa;
- Interface gráfica acessível, permitindo testes sem necessidade de programação aprofundada.

Esses recursos facilitam tanto a criação de testes simples quanto a execução de cenários mais complexos, sendo uma ferramenta ideal para iniciantes e profissionais experientes.

3.3. Instalação e Configuração

1. Acesse o site oficial: https://www.postman.com/downloads
2. Escolha a versão compatível com o seu sistema operacional (Windows, macOS ou Linux) e faça o download.

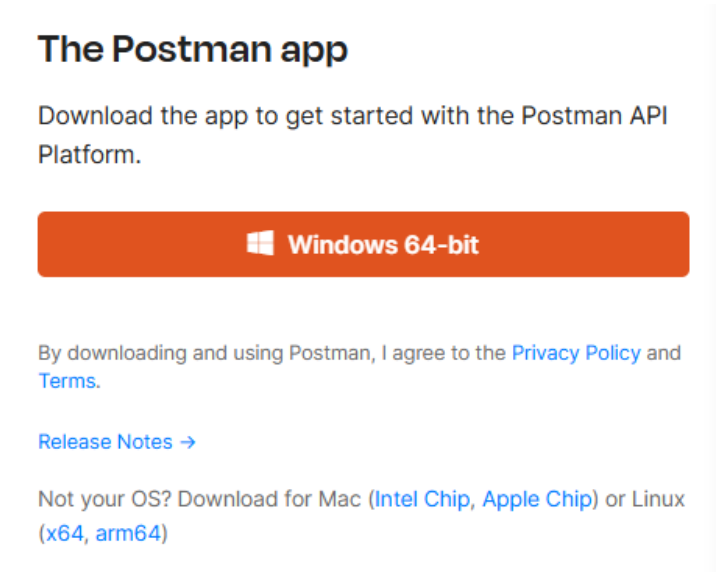

Figura 1 - Captura de tela da seção de downloads no website do Postman

Fonte: Website oficial do Postman.

3. Após o download, execute o instalador e aguarde a instalação completa.

Figura 2 - Captura de tela da janela de instalação do Postman

Fonte: Aplicativo do Postman para Windows.

4. Ao abrir o Postman, há a possibilidade de criar uma conta gratuita (opcional) para salvar coleções na nuvem. Basta clicar na opção “Sign up for Free” para ser redirecionado para o navegador para criar uma conta.

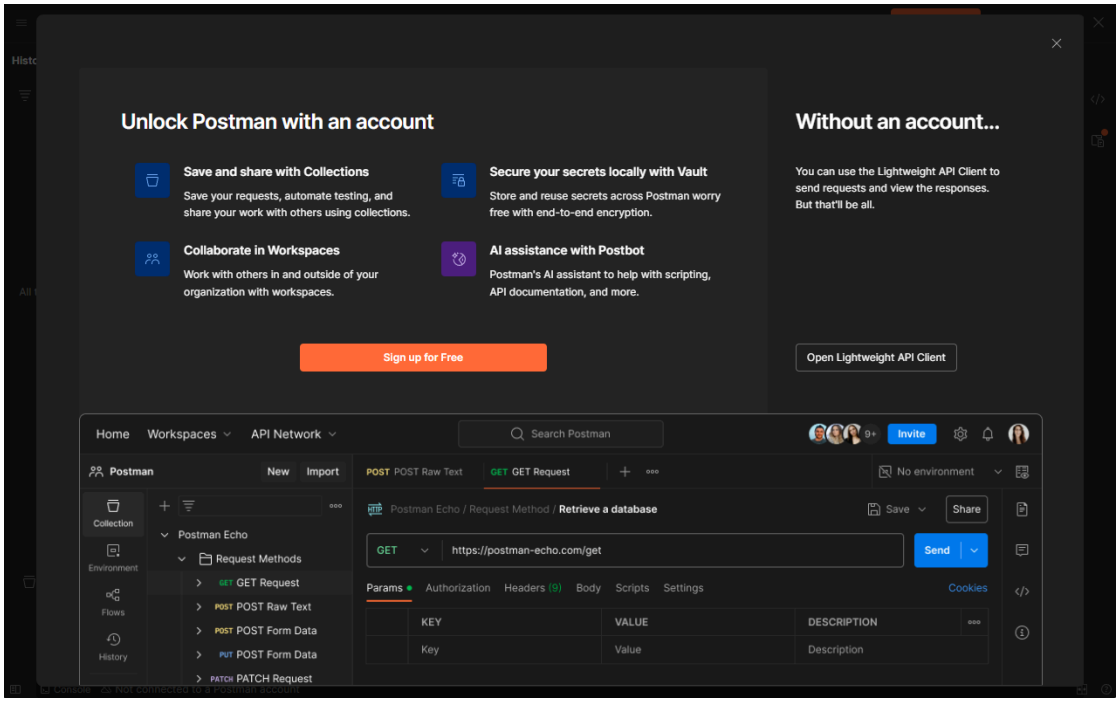

Figura 3 - Captura de tela inicial do aplicativo do Postman

Fonte: Aplicativo do Postman para Windows.

5. Para iniciar os testes, crie uma nova coleção clicando no botão “+” e depois em “Blank collection”.

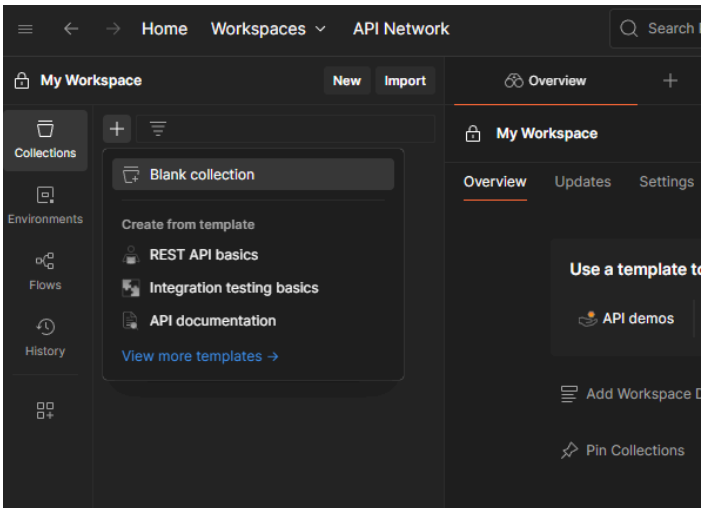

Figura 4 - Captura de tela da seção dedicada a criação de Collections

Fonte: Aplicativo do Postman para Windows.

6. Dentro da coleção, adicione requisições **GET**, **POST**, **PUT**, **DELETE** conforme os endpoints desejados.

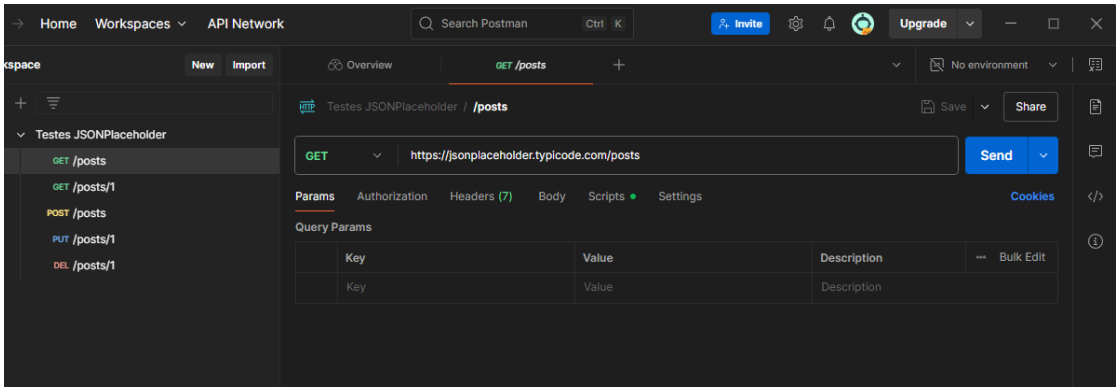

Figura 5 - Captura de tela da requisição GET /posts.
Fonte: Aplicativo do Postman para Windows.

7. Utilize a aba Tests para adicionar scripts de verificação automatizada.
8. Use a aba Body para enviar dados JSON quando necessário.
9. Para executar a requisição basta clicar no botão “Send” e receber a resposta logo abaixo.

A configuração adotada nesse projeto foi feita localmente, utilizando uma API pública de simulação, a JSONPlaceholder, que permite testar funcionalidades de CRUD em um ambiente seguro e controlado.

4. EXEMPLO PRÁTICO COM A FERRAMENTA

A realização dos testes automatizados com a ferramenta Postman acontecem sobre uma API REST simulada, disponibilizada publicamente pelo serviço JSONPlaceholder. A API escolhida é amplamente utilizada para fins educacionais e experimentais, por permitir operações de criação, leitura, atualização e exclusão (CRUD) de dados sem que alterações reais sejam persistidas no servidor.

4.1. Estrutura de Coleção de Testes

A coleção criada no Postman foi nomeada como **Testes_API_JSONPlaceholder**, e composta pelas seguintes requisições principais:

- `GET /posts` - Listagem de todos os posts;

- `GET /posts/{id}` - Consulta de um post específico.
- `POST /posts` - Criação de um novo post;
- `PUT /posts/{id}` - Atualização de um post existente;
- `DELETE /posts/{id}` - Exclusão de um post.

Cada requisição foi acompanhada de scripts de teste automatizado na aba Tests, com validação específica quanto ao código de status HTTP e ao conteúdo da resposta.

4.2. Etapas Realizadas

a) Criação de Requisições

Cada endpoint da API foi testado individualmente. Para os métodos POST e PUT, utilizou-se a aba Body do Postman com dados no formato JSON, como o exemplo abaixo:

```
{
"title": "Título de exemplo",
"body": "Conteúdo de teste",
"userId": 1
}
```

b) Execução dos Testes

Após configurar as requisições, cada uma foi executada com o botão Send, observando o tempo de resposta, o código HTTP retornado e os dados recebidos.

c) Automação com Scripts

Utilizaram-se scripts na aba Tests das requisições para validar automaticamente as respostas. Um exemplo desses scripts é o abaixo:

```
pm.test("Status code é 200", function () {
     pm.response.to.have.status(200);
});
```

Em requisições do tipo POST, também se validou o conteúdo do JSON retornado:

```
pm.test("Resposta contém título", function () {
      var jsonData = pm.response.json();
      pm.expect(jsonData).to.have.property("title");
});
```

d) Resultados

Os testes retornaram com sucesso, validando o comportamento esperado da API. Por exemplo, a requisição `POST /posts` respondeu com o status `201 Created` e retornou o objeto criado. Já a requisição `DELETE /posts/1` respondeu com `200 OK` ou `204 No Content`, indicando exclusão simulada bem-sucedida.

4.3. Capturas de Tela

As imagens abaixo ilustram a execução das requisições e os resultados dos testes automatizados no Postman:

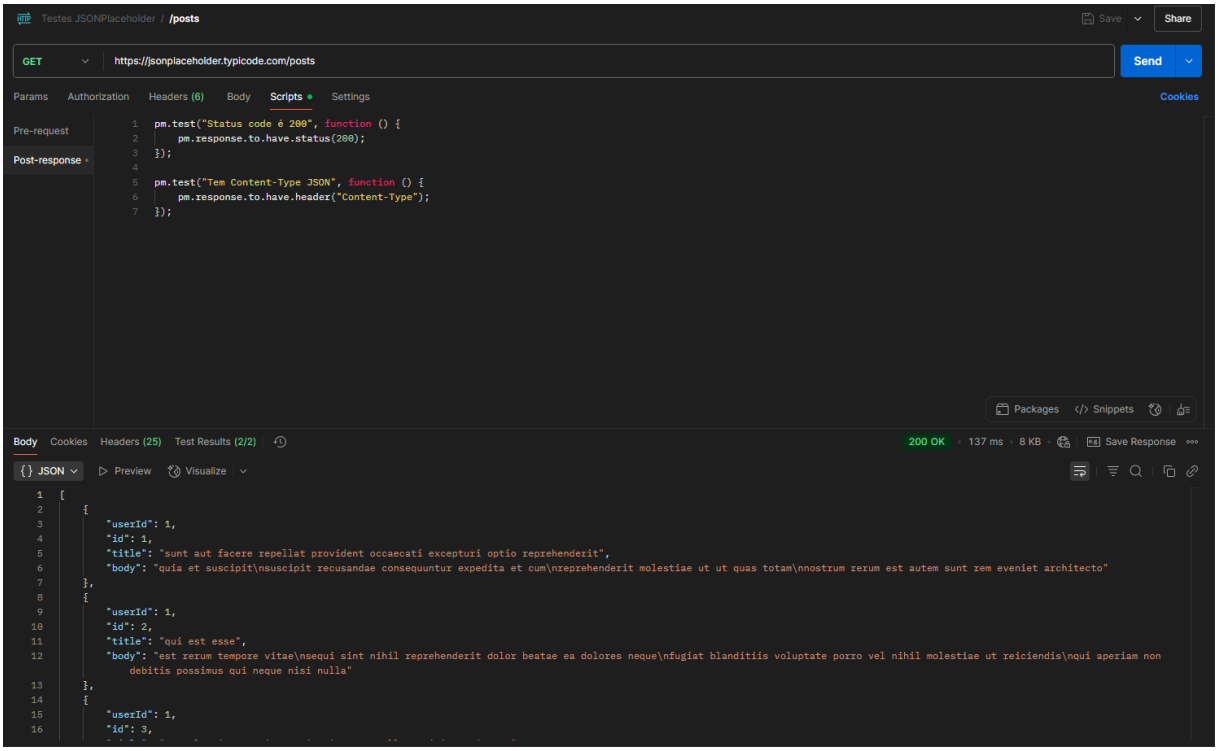

Figura 6 - Scripts e execução da requisição GET /posts.

Fonte: Aplicativo do Postman para Windows.

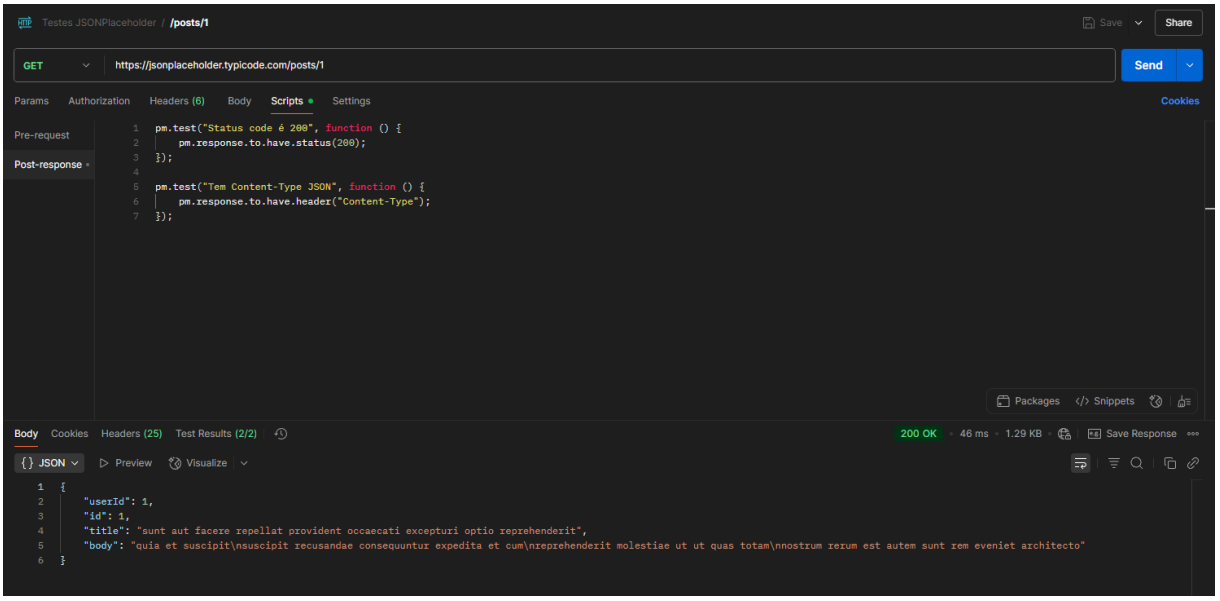

Figura 7 - Scripts e execução da requisição GET /posts/1.

Fonte: Aplicativo do Postman para Windows.

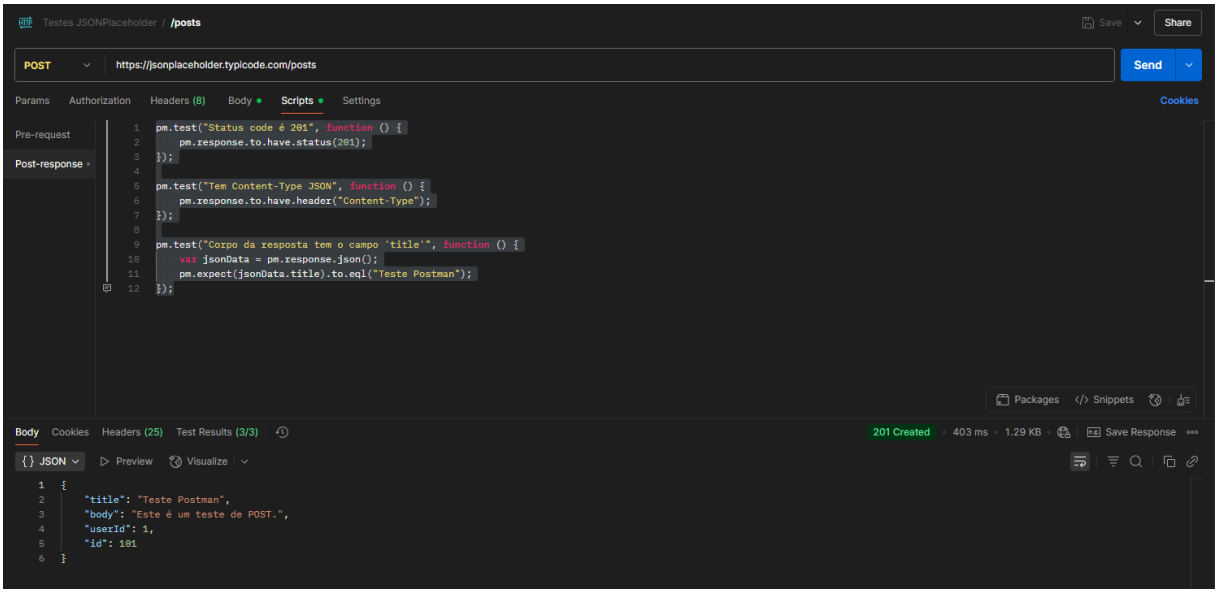

Figura 8 - Scripts e execução da requisição POST /posts.

Fonte: Aplicativo do Postman para Windows.

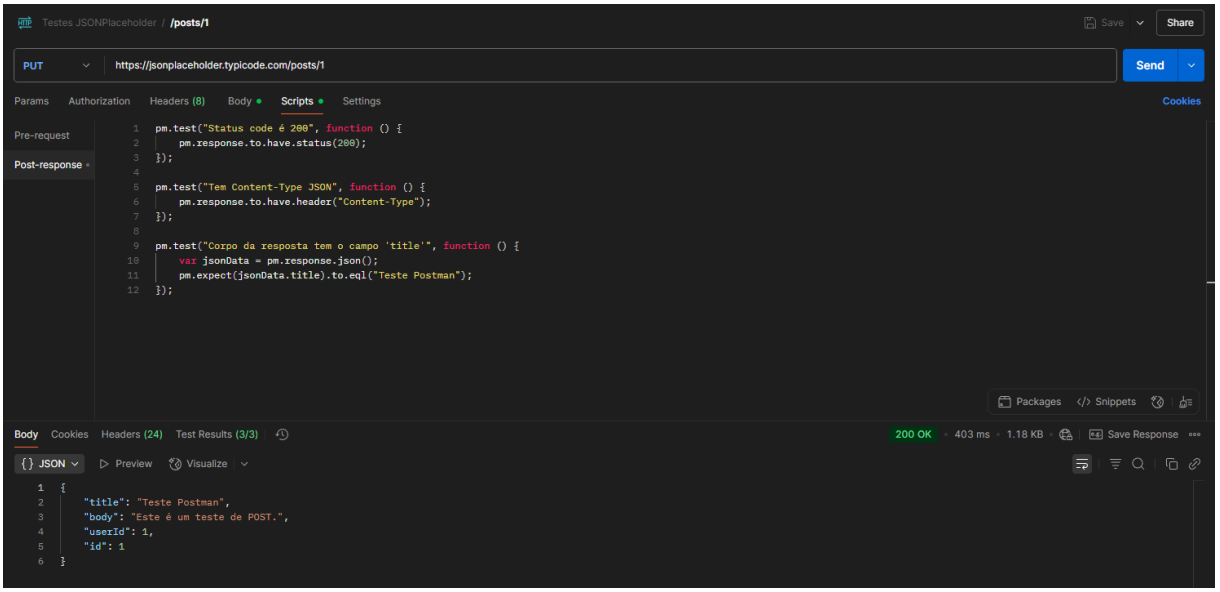

Figura 9 - Scripts e execução da requisição PUT /posts/1.

Fonte: Aplicativo do Postman para Windows.

Figura 10 - Scripts e execução da requisição DELETE /posts/1.

Fonte: Aplicativo do Postman para Windows.

5. ANÁLISE CRÍTICA DO POSTMAN

A partir da aplicação prática realizada na ferramenta Postman, é possível realizar uma análise crítica que contempla suas vantagens, limitações, a curva de aprendizado, usabilidade e os cenários ideais do uso. Essa avaliação contribui para

a escolha consciente da ferramenta em projetos reais, especialmente no contexto da garantia da qualidade de software.

5.1. Vantagens Identificadas

Durante a utilização do Postman, observou-se diversos pontos positivos:

- Interface intuitiva: o ambiente gráfico facilita a criação e execução de requisições HTTP, mesmo para iniciantes.
- Organização em coleções: permite estruturar testes por projetos ou funcionalidades, facilitando a manutenção.
- Suporte a scripts de teste: por meio de JavaScript, é possível automatizar verificações após cada requisição, aumentando a confiabilidade dos testes.
- Execução rápida de testes: a ferramenta responde rapidamente e permite simulações dinâmicas com variáveis e ambientes.
- Documentação automática: o Postman gera documentação das APIs testadas de maneira clara e acessível.
- Integração com CI/CD (na versão paga): a versão profissional permite integração direta com pipelines de integração contínua.

Essas características tornam o Postman uma ferramenta eficaz tanto para testes manuais quanto para automação básica de testes em APIs.

5.2. Limitações da Ferramenta

Apesar de suas funcionalidades robustas, algumas limitações foram observadas:

- Foco exclusivo em APIs: o Postman não é indicado para testes de interface gráfica ou testes de desempenho (stress ou carga).
- Limitações na automação em larga escala: para testes mais complexos ou em grande volume, ferramentas como JMeter ou scripts automatizados com frameworks de teste podem ser mais apropriadas.
- Necessidade de versão paga para recursos avançados: funcionalidades como monitoramento contínuo, múltiplos usuários colaborando em tempo real e integrações diretas com sistemas de CI/CD estão restritas a planos pagos.

- Dependência de conexão com a internet para alguns recursos, como sincronização com a conta em nuvem.

5.3. Curva de Aprendizado e Usabilidade

A curva de aprendizado do Postman pode ser considerada baixa, especialmente para testes simples. Usuários com conhecimentos básicos sobre o funcionamento de APIs REST conseguem utilizá-lo de forma eficaz com poucas horas de prática.

A usabilidade é um dos pontos fortes da ferramenta, com menus bem organizados, mensagens de erro compreensíveis e respostas visuais claras. Além disso, há vasta documentação e tutoriais disponíveis na internet, o que favorece seu uso no ambiente acadêmico e profissional.

5.4. Cenários Ideais de Uso

O postman é ideal para os seguintes contextos:

- Testes manuais e automatizados de APIs REST;
- Desenvolvimento e validação de endpoints durante a criação de sistemas;
- Simulação de dados e respostas de APIs durante o desenvolvimento frontend;
- Testes exploratórios durante a depuração de problemas em sistemas distribuídos;
- Treinamentos e estudos sobre consumo e integração com APIs.

Dessa forma, o Postman se consolida como uma ferramenta versátil, útil tanto para aprendizado quanto para aplicação em projetos reais.

6. CONCLUSÃO

A realização desta atividade prática, utilizando a ferramenta Postman para testes de APIs, permitiu consolidar conceitos fundamentais de qualidade de software e validação de sistemas. Através do estudo e aplicação de diferentes tipos de testes, em especial os testes de integração e sistema sobre uma API REST, foi possível

compreender na prática a importância da testabilidade e da automação no ciclo de desenvolvimento de software.

O Postman se mostrou uma ferramenta eficiente, acessível e bastante funcional para testes de APIs, mesmo em sua versão gratuita. Sua interface amigável, somada ao suporte a scripts de teste, contribuiu significativamente para a experiência de validação automatizada e análise de respostas HTTP. Além disso, a possibilidade de organizar requisições em coleções e utilizar variáveis e ambientes reforça sua aplicabilidade em projetos reais, colaborativos e escaláveis.

A atividade contribuiu diretamente para o aprendizado de práticas modernas de testes de software, alinhadas com a realidade do mercado, que cada vez mais exige profissionais capazes de garantir a confiabilidade, desempenho e segurança dos sistemas por meio de estratégias de teste bem definidas.

Por fim, destaca-se que o domínio de ferramentas como o Postman representa uma competência valiosa para o profissional da área de Engenharia de Qualidade, tanto em contextos de desenvolvimento quanto de testes e garantia de software, sendo aplicável em diversos tipos de projetos e ambientes de trabalho.

REFERÊNCIAS

POSTMAN. **Postman Learning Center: The Postman API Platform**. Postman. Disponível em: https://learning.postman.com. Acesso em: 19 mai. 2025.

POSTMAN. **Postman Documentation**. Postman. Disponível em: https://www.postman.com/product/api-client/. Acesso em: 19 mai. 2025.

EAD DOCS. **Testando a API com Postman**. EAD docs. Disponível em: https://docs.eadplataforma.com/docs/testando-a-api-com-postman. Acesso em: 19 mai. 2025.

DEVMEDIA. **Testando APIs Web com o Postman**. DevMedia. Disponível em: https://www.devmedia.com.br/testando-apis-web-com-o-postman/37264. Acesso em: 19 mai. 2025.