

✓ Code Review

📖 Introdução 🔗

- 🧑 Revisor: Raique Alfredo Pereira de Ramos
- 📅 Data: 23/05/2025
- 🧑 Autor do código: Thais do Amaral Cordeiro
- 🌐 API: ReqRes
- 🔗 Link do repositório: [aws-aiforsoftwarequalityengineering/Documentos/03. Entregas de Maio/robotframework at main · a-maralthx/aws-aiforsoftwarequalityengineering](https://github.com/maralthx/aws-aiforsoftwarequalityengineering/Documentos/03. Entregas de Maio/robotframework at main · a-maralthx/aws-aiforsoftwarequalityengineering)
- 🌿 Branch: main

☀️ Pontos Positivos 🔗

1. Clareza:

- 📖 O código está muito bem organizado e fácil de ler. A estrutura facilita a compreensão do fluxo e das funcionalidades implementadas. Parabéns por isso! 🎉

2. Boas Práticas:

- 📁 A separação do código em pastas e arquivos organizados, como a pasta `payloads` para JSONs, foi uma ótima decisão. Isso melhora a modularidade e a reutilização.
- ⚡ A utilização de guardar keywords nas variáveis para validar cenários é excelente. Essa abordagem torna o código mais enxuto e legível.

3. Inovação:

- 💡 Gostei da ideia de usar payloads separados para os Test Cases. Isso facilita alterações futuras e torna o código mais limpo.
- ✍️ A abordagem de manter a estrutura visual simples e eficiente nos arquivos de Test Cases e Keywords foi muito bem feita.

4. Cobertura de Testes:

- 🛠️ O projeto apresenta uma excelente cobertura de testes, contemplando de forma eficaz quatro endpoints distintos: GET, PUT, POST e PATCH. 🚀

🔍 Pontos de Melhoria 🔗

1. Leitura e Manutenção:

- 🖋️ **Comentários:** Seria útil adicionar comentários explicando o uso da biblioteca JSONLibrary. Isso ajuda a entender rapidamente a dependência e como configurá-la.
- 🛠️ **Erro ao reconhecer JSONLibrary:** Embora isso possa ser um problema local, vale a pena verificar a compatibilidade ou sugerir soluções na documentação do projeto.


2. Padrões e Consistência:

- 📄 O código está consistente com o restante da base, mas sugiro criar Test Cases para as seguintes Keywords:
 - 🗑️ **DELETE Usuário:** Isso garantiria a validação completa do endpoint e ajudaria a identificar problemas futuros.
 - 📝 **POST Registrar Usuário:** Um teste dedicado a esse endpoint validaria melhor os fluxos de registro.


3. Funcionalidade e Testes:


- 🔍 No cenário "PUT Atualizar Usuário Sucesso," não houve uma verificação explícita no código. Sugiro adicionar uma validação para garantir que a resposta inclua a propriedade "updatedAt."


4. Logs:

-  Adicionar mensagens personalizadas nos logs (por exemplo, "Usuário:") ajudaria a identificar rapidamente o conteúdo das respostas e reduz o tempo gasto para análise de problemas.
-

Conclusão

 O trabalho realizado neste código é admirável, destacando-se pela clareza, organização e inovação.

 A estrutura bem definida e o uso criativo de abordagens como variáveis nos Test Cases e a separação de payloads tornam o código um exemplo de boas práticas.

 Parabéns pelo excelente trabalho!! 