Code Review - API Reqres (Luis Magris)

Link do repositório onde foi realizado o code review:

compass_pb/Documentos/Sprint 5/robotframework/reqresAPI at f8d0786f06db446fdbbffe7c2c30a159b2cbc6e5 · luismagris/compass_pb

Code Review do arquivo 'api_testing_user.robot'

```
*** Settings ***
Resource  ../resources/api_testing_user.resource

*** Test Cases ***
Cenário: POST Fazer login
    Criar sessão no ReqRes
    POST Endpoint /login
    Validar status code "200"
```

'Criar sessão no ReqRes' é chamado no início de cada caso de teste. Já que essa condição é idêntica nos outros casos de testes, sugiro que utilize 'Suite Setup' ou 'Test Setup' no Robot Framework para evitar a repetição.
Exemplo:
*** Settings ***
Resource ../resources/api_testing_user.resource
Test Setup Criar sessão no ReqRes

*** Test Cases ***
Cenário: POST Fazer login
  POST Endpoint /login
  Validar status code "200"

... e assim por diante para os outros casos de teste

Palavras-chave como POST Endpoint /login são boas, mas você poderia considerar utilizar algo mais específico sobre o que está sendo enviado.
Por exemplo, se /login exige um nome de usuário e senha, sua palavra-chave pode ser: 'POST Login de Usuário Com Credenciais Válidas' ou 'POST Login de Usuário Sem Senha'
Vejo que isso pode tornar o caso de teste ainda mais auto-documentado sem a necessidade de verificar imediatamente o arquivo api_testing_user.resource.

```
Cenário: POST Fazer login UNSUCCESSFUL
    Criar sessão no ReqRes
    POST Endpoint /login - sem senha
    Validar status code "400"
```

Para os cenários "UNSUCCESSFUL", é bom que você utilize palavras-chave separadas como no 'POST Endpoint /login - sem senha'. Isso pode indicar mais claramente a condição específica para obter a falha.
Acredito que seria uma boa prática utilizar essas palavras-chave que realmente reflitam o payload / requisição "sem sucesso".

```
Cenário: POST Logout
    Criar sessão no ReqRes
    POST Endpoint /logout
    Validar status code "200"

Cenário: POST Cadastro de usuário
    [tags]    POST    user
    Criar sessão no ReqRes
    POST Endpoint /register
    Validar status code "200"
```

Excelente uso de tags! Sendo muito útil para realizar testes de forma mais seletiva.

```
Cenário: POST Cadastro de usuário UNSUCCESSFUL
    [tags]    POST    user
    Criar sessão no ReqRes
    POST Endpoint /register - sem senha
    Validar status code "400"

Cenário: GET Listar todos os usuários
    [Tags]    GET     user
    Criar sessão no ReqRes
    GET Endpoint /users
    Validar status code "200"

Cenário: GET Listar um usuário
    [Tags]    GET    user
    Criar sessão no ReqRes
    GET Endpoint /users/id
    Validar status code "200"

Cenário: PUT Editar um usuário
    [tags]    PUT    user
    Criar sessão no ReqRes
    PUT Endpoint /users/id
    Validar status code "200"

Cenário: PATCH Editar um usuário
    [Tags]    PATCH    user
    Criar sessão no ReqRes
    PATCH Endpoint /users/id
    Validar status code "200"

Cenário: DELETE Excluir um usuário
    [Tags]    DELETE    user
    Criar sessão no ReqRes
    DELETE Endpoint /users/id
    Validar status code "200"

Cenário: GET Listar um recurso
    [Tags]    GET    resource
    Criar sessão no ReqRes
    GET Endpoint /resource/id
    Validar status code "200"

Cenário: PUT Editar um recurso
    [Tags]    PUT    resource
```

```
    Criar sessão no ReqRes
    PUT Endpoint /resource/id
    Validar status code "200"

Cenário: DELETE Excluir um recurso
    [Tags]    DELETE    resource
    Criar sessão no ReqRes
    DELETE Endpoint /resource/id
    Validar status code "200"
```

Você possui várias chamadas que necessitam de um ID, e provavelmente seja espaço reservado. Quem sabe você não poderia utilizar um dos casos a seguir:

- Usar um ID estático conhecido: se você tiver um ambiente de teste onde pode garantir a existência de um ID específico.
- Criar um ID dinâmico: para requisições POST, frequentemente a resposta inclui o ID do recurso recém-criado. Capturando esse ID você poderá usá-lo em requisições GET, PUT, PATCH ou DELETE subsequentes dentro do mesmo fluxo de teste. Isso torna seus testes mais robustos e independentes de valores fixos.

```
Cenário: GET Listar todos os recursos
    [Tags]    GET    resource
    Criar sessão no ReqRes
    GET Endpoint /resource
    Validar status code "200"
```

Code Review do arquivo 'api_testing_user.resource'

```
*** Settings ***
Library                       String
Library                       RequestsLibrary

*** Keywords ***

POST Endpoint /login
    ${body}                   Create Dictionary
    ...                       email=eve.holt@reqres.in
    ...                       password=cityslicka
    ${response}               POST On Session
    ...                       alias=ReqRes
    ...                       url=/login
    ...                       json=${body}
    Set Global Variable       ${response}

POST Endpoint /login - sem senha
    ${body}                   Create Dictionary
    ...                       email=eve.holt@reqres.in
    ${response}               POST On Session
    ...                       alias=ReqRes
    ...                       url=/login
    ...                       json=${body}
    ...                       expected_status=400
    Set Global Variable       ${response}

POST Endpoint /logout
    ${response}               POST On Session
    ...                       alias=ReqRes
    ...                       url=/logout
```

```
    Set Global Variable      ${response}

Validar status code "${statuscode}"
    Should Be True           ${response.status_code} == ${statuscode}

```

As keywords 'POST Endpoint /login' e 'POST Endpoint /login - sem senha' (e igualmente para register) têm muito código em comum. Você poderia criar uma keyword auxiliar para a lógica de fazer a requisição POST, e passar o body e o status esperado como argumentos. Isso reduz a duplicação e torna o código mais fácil de manter. Exemplo:

```
POST Login Request
  [Arguments]   ${email}   ${password}=${NONE}   ${expected_status}=200
  ${body}=          Create Dictionary   email=${email}
  IF   '${password}' != '${NONE}'
    Set To Dictionary   ${body}   password=${password}
  END
  ${response}=        POST On Session
  ...               alias=ReqRes
  ...               url=/login
  ...               json=${body}
  Validar Status Code Da Resposta   ${response}   ${expected_status}
  RETURN            ${response}

POST Endpoint /login
  ${response}=        POST Login Request   eve.holt@reqres.in   cityslicka
  RETURN            ${response}

POST Endpoint /login - sem senha
  ${response}=        POST Login Request   eve.holt@reqres.in   ${NONE}   400
  RETURN            ${response}
```

Sugestão de validação da resposta:

```
POST Endpoint /login
  ${body}=          Create Dictionary   email=eve.holt@reqres.in   password=cityslicka
  ${response}=        POST On Session
  ...               alias=ReqRes
  ...               url=/login
  ...               json=${body}
  Validar Status Code Da Resposta   ${response}   200
  RETURN            ${response}
```

Em "POST Endpoint /login - sem senha", o uso de espaço-hífen-espaço (hífen duplo) é incomum. Deixo como sugestão utilizar um hífen simples (sem espaços) ou um underline, como por exemplo "POST Endpoint /login Sem Senha" ou "POST Endpoint /login_Sem_Senha". O mesmo se aplicará nas demais nomenclaturas semelhantes neste código.

"Validar status code" é uma boa nomenclatura, mas será que não poderia ser mais específico, como "Validar Status Code 'resposta esperada'", por exemplo?
Lembro também que a RequestsLibrary oferece muitas formas de validação, como por exemplo 'Should Be Status Code' ou 'Should Be Equal'

```
Criar sessão no ReqRes
    ${headers}              Create Dictionary
    ...                     accept=application/json
    ...                     Content-Type=application/json
```

```
    ...                     x-api-key=reqres-free-v1
    Create Session          ReqRes    https://reqres.in/api/  ${headers}

POST Endpoint /register
    ${body}                 Create Dictionary
    ...                     email=eve.holt@reqres.in
    ...                     password=pistol

    ${response}             POST On Session
    ...                     alias=ReqRes
    ...                     url=/register
    ...                     json=${body}
    Set Global Variable     ${response}

    Log                     ${response.json}

POST Endpoint /register - sem senha
    ${body}                 Create Dictionary
    ...                     email=sydney@fife

    ${response}             POST On Session
    ...                     alias=ReqRes
    ...                     url=/register
    ...                     json=${body}
    ...                     expected_status=400

    Set Global Variable     ${response}

    Log                     ${response.json}
```

'expected_status=400' é uma excelente forma para utilizar em validações rápidas de cenários de erro.

Atualmente, a variável ${response} está sendo definida como global em quase todas as keywords. Isso pode levar a problemas em cenários de teste mais complexos, onde você pode querer testar várias requisições em uma única execução ou em paralelo, e o valor de ${response} pode ser sobrescrito inesperadamente.
Considere fazer com que as keywords retornem o objeto de resposta. Assim, o teste (ou outra keyword) que chama essa keyword pode decidir o que fazer com a resposta. Por exemplo:
Cenário de Login Bem Sucedido
  Criar sessão no ReqRes
  ${resposta_login}= POST Endpoint /login
  Validar Status Code da Resposta ${resposta_login} 200

```
GET Endpoint /users

    ${response}             GET On Session
    ...                     alias=ReqRes
    ...                     url=/users
    Set Global Variable     ${response}
    Log                     ${response.content}

GET Endpoint /users/id
    ${response}             GET On Session
    ...                     alias=ReqRes
    ...                     url=/users/1
    Set Global Variable     ${response}
    Log                     ${response.content}
```

```
PUT Endpoint /users/id
    ${payload}              Create Dictionary
    ...                     name=ze
    ...                     job=pedreiro
    ${response}             PUT On Session
    ...                     alias=ReqRes
    ...                     url=/users/3
    ...                     json=${payload}
    Set Global Variable     ${response}

    Log                     ${response.content}

```

Você está usando PUT On Session para o endpoint PATCH. Isso está incorreto e enviará uma requisição PUT em vez de PATCH. Corrija para PATCH On Session.

```
PATCH Endpoint /users/id
    ${payload}              Create Dictionary
    ...                     name=joao
    ...                     job=jogador
    ${response}             PUT On Session
    ...                     alias=ReqRes
    ...                     url=/users/5
    ...                     json=${payload}
    Set Global Variable     ${response}

    Log                     ${response.content}

DELETE Endpoint /users/id
    ${response}             PUT On Session
    ...                     alias=ReqRes
    ...                     url=/users/5
    Set Global Variable     ${response}

    Log                     ${response.content}
```

Você está usando PUT On Session para o endpoint DELETE. Isso está incorreto e enviará uma requisição PUT em vez de DELETE. Corrija para DELETE On Session.

```
GET Endpoint /resource/id
    ${response}             GET On Session
    ...                     alias=ReqRes
    ...                     url=/unknown/2
    Set Global Variable     ${response}
    Log                     ${response.content}

GET Endpoint /resource
    ${response}             GET On Session
    ...                     alias=ReqRes
    ...                     url=/unknown
    Set Global Variable     ${response}
    Log                     ${response.content}

PUT Endpoint /resource/id
    ${payload}              Create Dictionary
    ...                     name=azul
```

```
    ...                     year=2000
    ...                     color=#C48394
    ${response}             PUT On Session
    ...                     alias=ReqRes
    ...                     url=/unknown/2
    ...                     json=${payload}
    Set Global Variable     ${response}

    Log                     ${response.content}

DELETE Endpoint /resource/id
    ${response}             GET On Session
    ...                     alias=ReqRes
    ...                     url=/unknown/3
    Set Global Variable     ${response}
    Log                     ${response.content}
```

Segue uma sugestão de código refatorado:

```
*** Settings ***
Library       String
Library       RequestsLibrary

*** Keywords ***

**--- Keywords de Reuso ---

Criar Sessao ReqRes
  ${headers}=          Create Dictionary    accept=application/json    Content-Type=application/json
  # Remover x-api-key se não for estritamente necessário para Reqres - A hosted REST-API ready to respond to your AJAX requests
  # ${headers}=          Create Dictionary    accept=application/json    Content-Type=application/json    x-api-key=reqres-free-v1
  Create Session       ReqRes           https://reqres.in/api/    headers=${headers}

Validar Status Code Da Resposta
  [Arguments]    ${response_obj}    ${expected_status_code}
  Should Be Equal    ${response_obj.status_code}    ${expected_status_code}
  Log To Console    Status code esperado: ${expected_status_code}, Status code recebido: ${response_obj.status_code}

Logar Conteudo Da Resposta
  [Arguments]    ${response_obj}
  Log    ${response_obj.content}

**--- Keywords de Endpoint ---

POST Login Request
  [Arguments]    ${email}    ${password}=${NONE}    ${expected_status}=200
  ${body}=             Create Dictionary    email=${email}
  IF    '${password}' != '${NONE}'
    Set To Dictionary    ${body}    password=${password}
  END
  ${response}=         POST On Session
  ...                  alias=ReqRes
  ...                  url=/login
  ...                  json=${body}
  Validar Status Code Da Resposta    ${response}    ${expected_status}
```

```
    Logar Conteudo Da Resposta    ${response}
    RETURN              ${response}

POST Register Request
    [Arguments]    ${email}    ${password}=${NONE}    ${expected_status}=200
    ${body}=              Create Dictionary    email=${email}
    IF    '${password}' != '${NONE}'
        Set To Dictionary    ${body}    password=${password}
    END
    ${response}=          POST On Session
    ...                   alias=ReqRes
    ...                   url=/register
    ...                   json=${body}
    Validar Status Code Da Resposta    ${response}    ${expected_status}
    Logar Conteudo Da Resposta    ${response}
    RETURN              ${response}

PUT ou PATCH User Request
    [Arguments]    ${method}    ${user_id}    ${name}    ${job}    ${expected_status}=200
    ${payload}=           Create Dictionary    name=${name}    job=${job}
    ${response}=          ${method} On Session  # Permite passar PUT On Session ou PATCH On Session
    ...                   alias=ReqRes
    ...                   url=/users/${user_id}
    ...                   json=${payload}
    Validar Status Code Da Resposta    ${response}    ${expected_status}
    Logar Conteudo Da Resposta    ${response}
    RETURN              ${response}

PUT Resource Request
    [Arguments]    ${resource_id}    ${name}    ${year}    ${color}    ${expected_status}=200
    ${payload}=           Create Dictionary    name=${name}    year=${year}    color=${color}
    ${response}=          PUT On Session
    ...                   alias=ReqRes
    ...                   url=/unknown/${resource_id}
    ...                   json=${payload}
    Validar Status Code Da Resposta    ${response}    ${expected_status}
    Logar Conteudo Da Resposta    ${response}
    RETURN              ${response}

** --- Chamadas Específicas de Endpoint (usando as keywords de reuso) ---

POST Endpoint /login
    ${response}=          POST Login Request    eve.holt@reqres.in    cityslicka
    RETURN              ${response}

POST Endpoint /login Sem Senha
    ${response}=          POST Login Request    eve.holt@reqres.in    ${NONE}    400
    RETURN              ${response}

POST Endpoint /logout
    ${response}=          POST On Session
    ...                   alias=ReqRes
    ...                   url=/logout
    Validar Status Code Da Resposta    ${response}    200 # Assumindo 200 OK para logout
```

```
    Logar Conteudo Da Resposta    ${response}
    RETURN              ${response}

POST Endpoint /register
    ${response}=          POST Register Request    eve.holt@reqres.in    pistol
    RETURN              ${response}

POST Endpoint /register Sem Senha
    ${response}=          POST Register Request    sydney@fife    ${NONE}    400
    RETURN              ${response}

GET Endpoint /users
    ${response}=          GET On Session
    ...                 alias=ReqRes
    ...                 url=/users
    Validar Status Code Da Resposta    ${response}    200
    Logar Conteudo Da Resposta    ${response}
    RETURN              ${response}

GET Endpoint /users/id
    [Arguments]    ${user_id}
    ${response}=          GET On Session
    ...                 alias=ReqRes
    ...                 url=/users/${user_id}
    Validar Status Code Da Resposta    ${response}    200
    Logar Conteudo Da Resposta    ${response}
    RETURN              ${response}

PUT Endpoint /users/id
    [Arguments]    ${user_id}    ${name}    ${job}
    ${response}=          PUT ou PATCH User Request    PUT    ${user_id}    ${name}    ${job}
    RETURN              ${response}

PATCH Endpoint /users/id
    [Arguments]    ${user_id}    ${name}    ${job}
    ${response}=          PUT ou PATCH User Request    PATCH    ${user_id}    ${name}    ${job}
    RETURN              ${response}

DELETE Endpoint /users/id
    [Arguments]    ${user_id}
    ${response}=          DELETE On Session # Corrigido de PUT para DELETE
    ...                 alias=ReqRes
    ...                 url=/users/${user_id}
    Validar Status Code Da Resposta    ${response}    204 # Assumindo 204 No Content para DELETE bem-sucedido
    Logar Conteudo Da Resposta    ${response}
    RETURN              ${response}

GET Endpoint /resource/id
    [Arguments]    ${resource_id}
    ${response}=          GET On Session
    ...                 alias=ReqRes
    ...                 url=/unknown/${resource_id}
    Validar Status Code Da Resposta    ${response}    200
    Logar Conteudo Da Resposta    ${response}
    RETURN              ${response}
```

```
GET Endpoint /resource
    ${response}=          GET On Session
    ...                   alias=ReqRes
    ...                   url=/unknown
    Validar Status Code Da Resposta    ${response}    200
    Logar Conteudo Da Resposta    ${response}
    RETURN                ${response}

PUT Endpoint /resource/id
    [Arguments]    ${resource_id}    ${name}    ${year}    ${color}
    ${response}=          PUT Resource Request    ${resource_id}    ${name}    ${year}    ${color}
    RETURN                ${response}

DELETE Endpoint /resource/id
    [Arguments]    ${resource_id}
    ${response}=          DELETE On Session # Corrigido de GET para DELETE
    ...                   alias=ReqRes
    ...                   url=/unknown/${resource_id}
    Validar Status Code Da Resposta    ${response}    204 # Assumindo 204 No Content para DELETE bem-sucedido
    Logar Conteudo Da Resposta    ${response}
    RETURN                ${response}
```