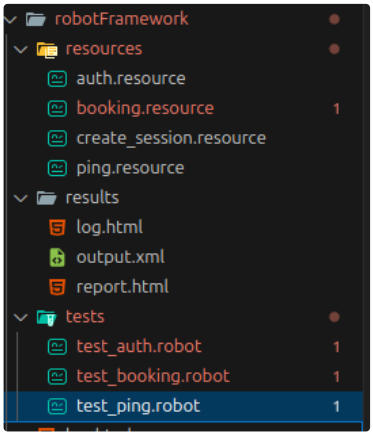
Code Review Cássia

Tarefas:

1. Nome de quem fez o CodeReview—> Douglas Paulo Cortes
2. Nome do colega que criou o código—> Cássia Basso
3. Link do repositório e qual branch foi avaliada (preferencialmente main, com merge feito pelo colega)—> PB_Compass/entregas/sprint5/robotFramework at main · cassiab13/PB_Compass
4. Resultado do Codereview, pontos levantados e o que foi levantado como ponto de melhoria no código—> Segue a avaliação abaixo

1. Organização e Estrutura:

- **Pastas claras e intuitivas, espelhando a API**



- **Uso de Keywords Reutilizáveis:**

```
*** Keywords ***
Criar um novo booking
    ${bookingdates}    Create Dictionary    checkin=2025-05-01    checkout=2025-05-05

    ${payload}    Create Dictionary
    ...           firstname=Johnny
    ...           lastname=Brown
    ...           totalprice=${100}
    ...           depositpaid=${False}
    ...           bookingdates=${bookingdates}
    ...           additionalneeds=Breakfast

    ${response}    POST On Session    RestfulBooker    /booking    json=${payload}
    Set Global Variable    ${booking_id}    ${response.json()["bookingid"]}
Fazer o update de um Booking
    [Arguments]    ${status_code_desejado}

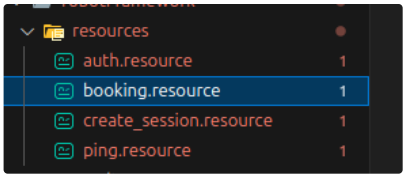
    ${cookies}    Create Dictionary      Cookie=token=${token}
    ${bookingdates}    Create Dictionary    checkin=2025-05-01    checkout=2025-05-05

    ${updated_payload}    Create Dictionary
    ...         firstname=James
    ...         lastname=Brown
    ...         totalprice=${100}
    ...         depositpaid=${True}
    ...         bookingdates=${booking_dates}
    ...         additionalneeds=Breakfast

    ${auth_header}    Create Dictionary    Cookie=token=${token}
    ${response_update}    PUT On Session    RestfulBooker    /booking/${booking_id}
```

-

2. **Criação de uma camada de abstração**:

- 

- Isso facilita manutenção e **reutilização** em outros testes.

---

3. Parametrização de dados:

- Os d**ados de payloads** (exemplo: `token`) foram extraídos:
  - ```
    *** Keywords ***
    Realizar o login como administrador
        [Arguments]  ${status_code_desejado}
        ${body}  Create Dictionary
        ...      username=admin
        ...      password=password123

        ${response}  POST On Session
        ...          alias=RestfulBooker
        ...          url=/auth
        ...          json=${body}
        ...          expected_status=${status_code_desejado}

        Set Global Variable    ${token}    ${response.json()["token"]}

    Verificar se token foi gerado
    ```

Vantagem: facilita manutenção e permite gerar dados dinâmicos para testes com múltiplos cenários.

---

4. Uso de Tags:

- Vejo uso de tags como `PUTBooking`, `PATCHBooking` — muito bom!

Seria interessante também adicionar tags por **prioridade** ou **tipo de teste**, por exemplo:

- `@sanity`
- `@regression`
- `@critical`

Assim, podem executar subsets de testes facilmente:

`robot --include regression tests/`

5. Cobertura:

Endpoint	Método	Status esperado	Testado?	Comentário
`/auth`	POST	200	✅ Sim	Testado, passou
`/auth`	POST	401	❌ Não	**Não há teste para autenticação inválida**
`/booking`	PUT	200	✅ Sim	Atualização completa testada
`/booking`	GET	200	✅ Sim	Listagem testada
    ```

`/booking`	PATCH	200	✅ Sim	Atualização parcial testada
`/booking`	DELETE	201 (ou 204)	✅ Sim	Testado, com verificação de inexistência posterior
`/booking`	POST	200	✅ Sim	Implícito, pois usado para preparar dados em PUT, PATCH e DELETE
`/ping`	GET	201	✅ Sim	Testado, mas documentação sugere código 201? Normalmente 200

❗ Pontos de melhoria (oportunidades):

✅ /auth:

- ✅ Testado com **sucesso** (200 OK).
- ⚠️ **Falta testar**:
  - Cenário de **falha** com credenciais inválidas (**401 Unauthorized**).

✅ /booking:

- ✅ Testado com sucesso para todos os métodos principais.
- ❓ **Faltou** verificar:
  - Respostas de **erro** ou **validações**:
    - Ex.: criar com payload inválido → resposta 400.
    - PUT, PATCH, DELETE para **IDs inexistentes** → resposta 404.
  - GET com **filtros** → ex.: `GET /booking?firstname=Mary`.

Relatório da branch adicionada

Criando um cenário negativo no teste, com credenciais inválidas, temos Status Code 200, indicando uma falha da API.

```
18
19  Realizar login com credenciais inválidas
20      [Arguments]  ${status_code_desejado}
21      ${body}  Create Dictionary
22      ...      username=admin
23      ...      password=senha_errada
24
25      ${response}  POST On Session
26      ...          alias=RestfulBooker
27      ...          url=/auth
28      ...          json=${body}
29      ...          expected_status=${status_code_desejado}
30
31      Set Global Variable    ${token}    ${response.json()["token"]}
32
33  Verificar se token foi gerado
```

```
Cenario 02: POST /auth 401
    [Tags]    POSTAuth
    Criar Sessão no Restful Booker
    Realizar login com credenciais inválidas    status_code_desejado=401
    Verificar se token foi gerado
```

```
TEST Cenario 02: POST /auth 401
Full Name:          Tests.Test Auth.Cenario 02: POST /auth 401
Tags:               POSTAuth
Start / End / Elapsed:  20250523 16:21:37.008 / 20250523 16:21:37.669 / 00:00:00.661
Status:             FAIL
Message:            Url: https://restful-booker.herokuapp.com/auth Expected status: 200 != 401
KEYWORD create_session. Criar Sessão no Restful Booker
KEYWORD auth. Realizar login com credenciais inválidas  status_code_desejado=401
  Start / End / Elapsed:  20250523 16:21:37.013 / 20250523 16:21:37.668 / 00:00:00.655
  KEYWORD ${body} = BuiltIn. Create Dictionary  username=admin  password=senha_errada
  KEYWORD ${response} = RequestsLibrary. POST On Session  alias=RestfulBooker  url=/auth  json=${body}  expected_status=${status_code_desejado}
    Documentation:  Sends a POST request on a previously created HTTP Session.
```

Incrementando um PAYLOAD inválido, temos um coportamento não tratado da API, com erro de servidor(Status Code 500

```
88
89  Criar booking com payload inválido
90      [Arguments]    ${status_code_desejado}
91      ${payload_invalido}=    Create Dictionary    firstname=    lastname=
92      ${response}=    POST On Session
93      ...    alias=RestfulBooker
94      ...    url=/booking
95      ...    json=${payload_invalido}
96      ...    expected_status=${status_code_desejado}
97      Set Suite Variable    ${response}
98
99  Verificar resposta de erro para payload inválido
100     Should Be Equal As Integers    ${response.status_code}    400
101     Log    ${response.json()}
```

```
Cenario 05: POST /booking com payload inválido 400
    [Tags]    POSTBooking    Negativo
    Criar Sessão no Restful Booker
    Criar booking com payload inválido    status_code_desejado=400
    Verificar resposta de erro para payload inválido
```

```
TEST Cenario 05: POST /booking com payload inválido 400
Full Name:          Tests.Test Booking.Cenario 05: POST /booking com payload inválido 400
Tags:               Negativo, POSTBooking
Start / End / Elapsed:  20250523 16:35:01.830 / 20250523 16:35:02.488 / 00:00:00.658
Status:             FAIL
Message:            Url: https://restful-booker.herokuapp.com/booking Expected status: 500 != 400
KEYWORD create_session. Criar Sessão no Restful Booker
KEYWORD booking. Criar booking com payload inválido  status_code_desejado=400
  Start / End / Elapsed:  20250523 16:35:01.836 / 20250523 16:35:02.487 / 00:00:00.651
  KEYWORD ${payload_invalido} = BuiltIn. Create Dictionary  firstname=  lastname=
  KEYWORD ${response} = RequestsLibrary. POST On Session  alias=RestfulBooker  url=/booking  json=${payload_invalido}  expected_status=${status_code_desejado}
    Documentation:  Sends a POST request on a previously created HTTP Session.
    Start / End / Elapsed:  20250523 16:35:01.838 / 20250523 16:35:02.486 / 00:00:00.648
    16:35:02.484  INFO  POST Request : url=https://restful-booker.herokuapp.com/booking
```