Módulo 5 – Requisições, Headers e Autenticação

Objetivo do módulo

Aprender a:

- Enviar diferentes tipos de requisições HTTP (`GET`, `POST`, `PUT`, `DELETE`)
- Utilizar **headers** e **payloads**
- Autenticar usuários com **JWT (JSON Web Token)**
- Validar respostas utilizando o módulo `check` do k6

---

1. Requisição GET

A requisição `GET` é utilizada para buscar dados de um servidor.

```
import http from 'k6/http';

export default function () {
  const res = http.get('http://localhost:3000/produtos');
  console.log(res.status, res.body.substring(0, 100));
}
```

Explicação detalhada

- `import http from 'k6/http'`: importa o módulo responsável por requisições HTTP.
- `http.get(url)`: executa uma requisição do tipo GET para a URL informada.
- `res.status`: exibe o código de status da resposta, como `200` (OK), `404` (não encontrado), `500` (erro interno).
- `res.body`: contém o corpo da resposta, geralmente em formato JSON.
- `substring(0, 100)`: corta o texto para mostrar apenas os 100 primeiros caracteres do corpo no console.

Exemplo de resposta real

Se o endpoint `/produtos` retornar uma lista de produtos, o `res.body` pode conter algo como:

```
[
  {
    "id": "p123",
    "nome": "Notebook Dell",
    "preco": 4200,
    "estoque": 12
  },
  {
    "id": "p124",
    "nome": "Mouse sem fio",
    "preco": 120,
    "estoque": 45
  }
]
```

---

2. Requisição POST

A requisição `POST` é utilizada para criar novos recursos no servidor, enviando dados no corpo da requisição.

```
import http from 'k6/http';

export default function () {
  const payload = JSON.stringify({
    nome: 'Produto Teste',
    preco: 120,
    descricao: 'Produto criado via teste automatizado',
    quantidade: 10,
  });

  const headers = { 'Content-Type': 'application/json' };

  const res = http.post('http://localhost:3000/produtos', payload, { headers
});
```

```
  console.log(`Status: ${res.status}`);
}
```

Explicação detalhada

- `JSON.stringify()`: converte o objeto JavaScript em uma string JSON.
- `payload`: representa os dados que serão enviados no corpo da requisição.
- `headers`: define os cabeçalhos da requisição; o `Content-Type` informa ao servidor que o conteúdo é JSON.
- `http.post(url, payload, { headers })`: envia o corpo JSON para o endpoint informado.
- `res.status`: mostra o código de status retornado pelo servidor.
  - `201`: criado com sucesso
  - `400`: requisição inválida
  - `500`: erro interno no servidor

Exemplo de resposta real

```
{
  "message": "Produto criado com sucesso",
  "id": "p999"
}
```

---

3. Requisições PUT e DELETE

O método `PUT` é usado para atualizar dados já existentes.

O método `DELETE` é usado para remover registros do servidor.

```
import http from 'k6/http';

export default function () {
  const url = 'http://localhost:3000/produtos/p999';
  const payload = JSON.stringify({ preco: 200 });

  http.put(url, payload, { headers: { 'Content-Type': 'application/json' } });
```

```
  http.del(url);
}
```

Explicação detalhada

- `PUT`: atualiza um registro existente com base no ID informado na URL.
- `DELETE`: remove um registro específico.
- `payload`: define os novos dados para atualização.
- `headers`: garante que o conteúdo está sendo enviado como JSON.
- `http.put()` e `http.del()` são chamados separadamente, simulando a atualização e remoção de um item.

Exemplo de fluxo real

1. `PUT /produtos/p999` → Atualiza o preço do produto para `200`.
2. `DELETE /produtos/p999` → Remove o produto `p999` do banco.
