

# JWT

JSON Web Tokens

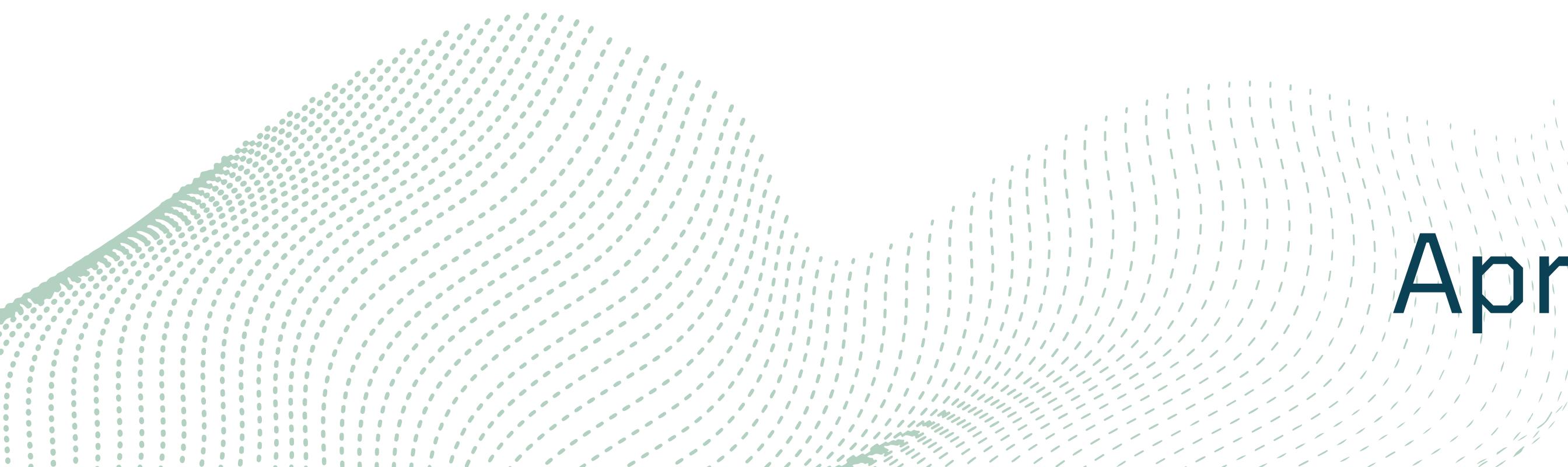


**Estágio Orientado à Docência**



# **Eduardo Luiz Schade Soares**

Programa de Pós-Graduação em Computação Aplicada

A decorative graphic at the bottom left consists of two overlapping, semi-transparent greenish-blue curved bands. The upper band has a dotted pattern, while the lower band has a fine grid pattern.

**Apresentação**



# O que é Segurança de Software?

- Confidencialidade
- Integridade
- Funcionalidade

# Segurança de Software é...

a ideia de projetar um Software que funcione  
plenamente mesmo sob ataques maliciosos.

MCGRAW (2004)

# Regulamentação



**GDPR (2018)**

General Data Protection Regulations



**LGPD (2020)**

Lei Geral de Proteção de Dados

# Regulamentação

A aplicação deve garantir que suas informações não sejam acessadas, alteradas ou interceptadas. Nenhuma parte além dos usuários a quem esses dados se destinam deve ser capaz de interagir de qualquer forma com eles.

SPILCA (2020)

# Uma aplicação segura



Quanto melhor cada camada for protegida

menor a chance de um indivíduo mal-intencionado conseguir acessar os dados ou realizar operações não autorizadas.

# Autenticação vs Autorização



Error

This password is already used by starboy98. Try another.

OK

**starboy98**



# Qual a diferença?

## Autenticação

- Representa o **processo** no qual um aplicativo **identifica** alguém tentando usá-lo;
- Identifica um **usuário** (uma **pessoa** ou **aplicativo**);
- Após a identificação, pode **decidir** o que lhes deve ser **permitido** fazer.

## Autorização

- Representa o processo de **estabelecer o acesso** de um usuário autenticado. Verificando se o mesmo **possui privilégios** para usar funcionalidades e dados específicos;
- Uma autorização está **quebrada** se um usuário obtém acesso a uma funcionalidade ou dados que **não pertencem a ele**.

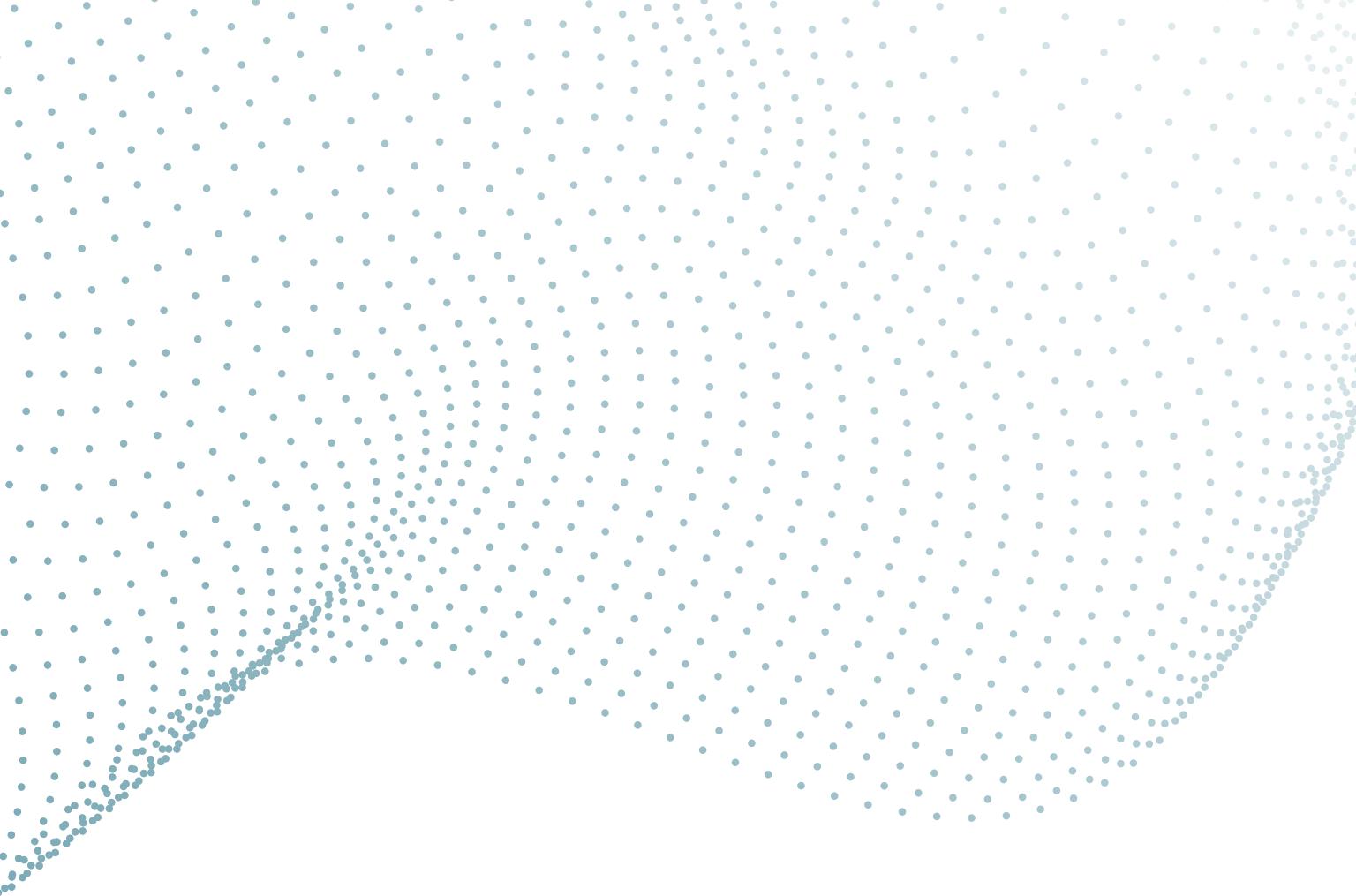
# JSON Web Token

# O que é JWT?

JWT significa **JSON Web Token** e é um **padrão de autorização** de usuário **sem estado (stateless)** comumente usado para transmitir informações com segurança entre cliente e servidor em um formato **JSON**

Por padrão, JWT é **codificado e não criptografado**.

Ele é **assinado** digitalmente usando uma **chave secreta** conhecida **apenas pelo servidor**.

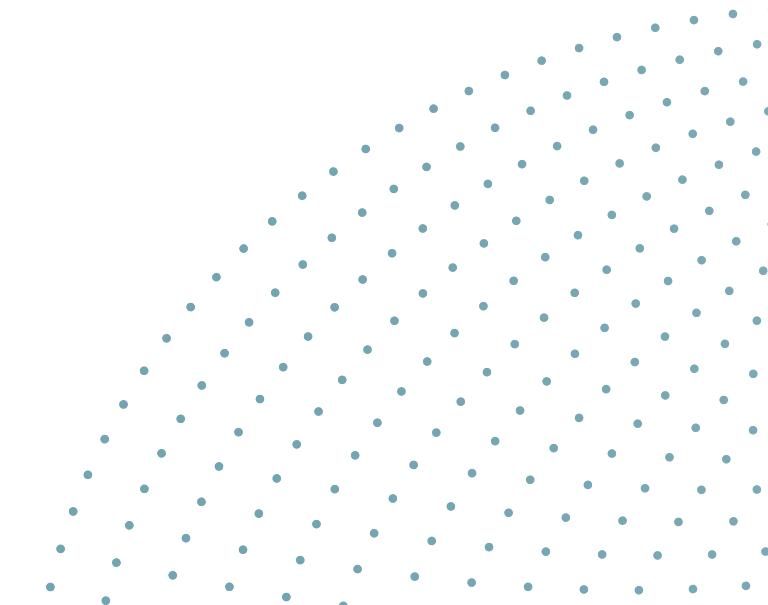


# JSON

## JavaScript Object Notation

É um formato **leve** de troca de dados. É fácil para humanos **larem e escreverem**. É fácil para máquinas **analisarem e gerarem**. É baseado em um subconjunto do JavaScript Programming Language Standard ECMA-262 3<sup>a</sup> edição - dezembro de 1999.

JSON é um formato de texto que é **completamente independente** de **linguagem**, mas usa convenções que são familiares para programadores da família de linguagens C, incluindo C, C++, C#, Java, JavaScript, Perl, Python e muitas outras. Essas propriedades tornam JSON uma linguagem **ideal de troca de dados**.



```
{  
  "usuario": {  
    "id": 12345,  
    "nome": "João Silva",  
    "email": "joao.silva@example.com",  
    "idade": 28,  
    "endereco": {  
      "rua": "Av. Principal",  
      "numero": 456,  
      "cidade": "São Paulo",  
      "estado": "SP",  
      "cep": "12345-678"  
    },  
    "telefones": [  
      {  
        "tipo": "celular",  
        "numero": "(11) 98765-4321"  
      },  
      {  
        "tipo": "fixo",  
        "numero": "(11) 3456-7890"  
      }  
    ]  
  }  
}
```

# JSON

## JavaScript Object Notation

JSON organiza os dados em **pares de chave-valor**. Cada chave é uma string (envolta em aspas) e está associada a um valor, que pode ser um número, string, booleano, objeto, array ou null.

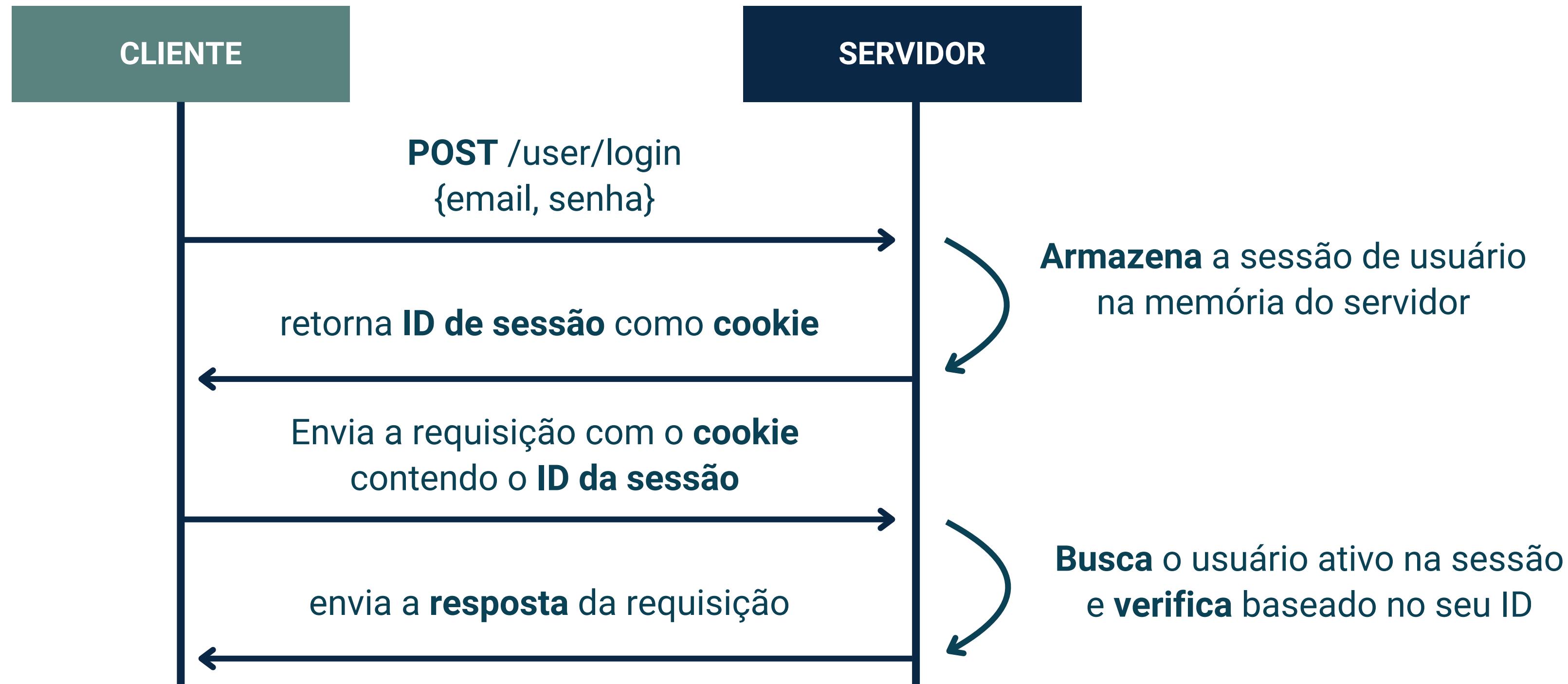
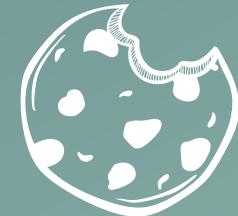
**Objetos:** Representados por chaves { }, contêm pares de chave-valor.

**Arrays:** Representados por colchetes [ ], contêm uma lista de valores, que podem ser de qualquer tipo.

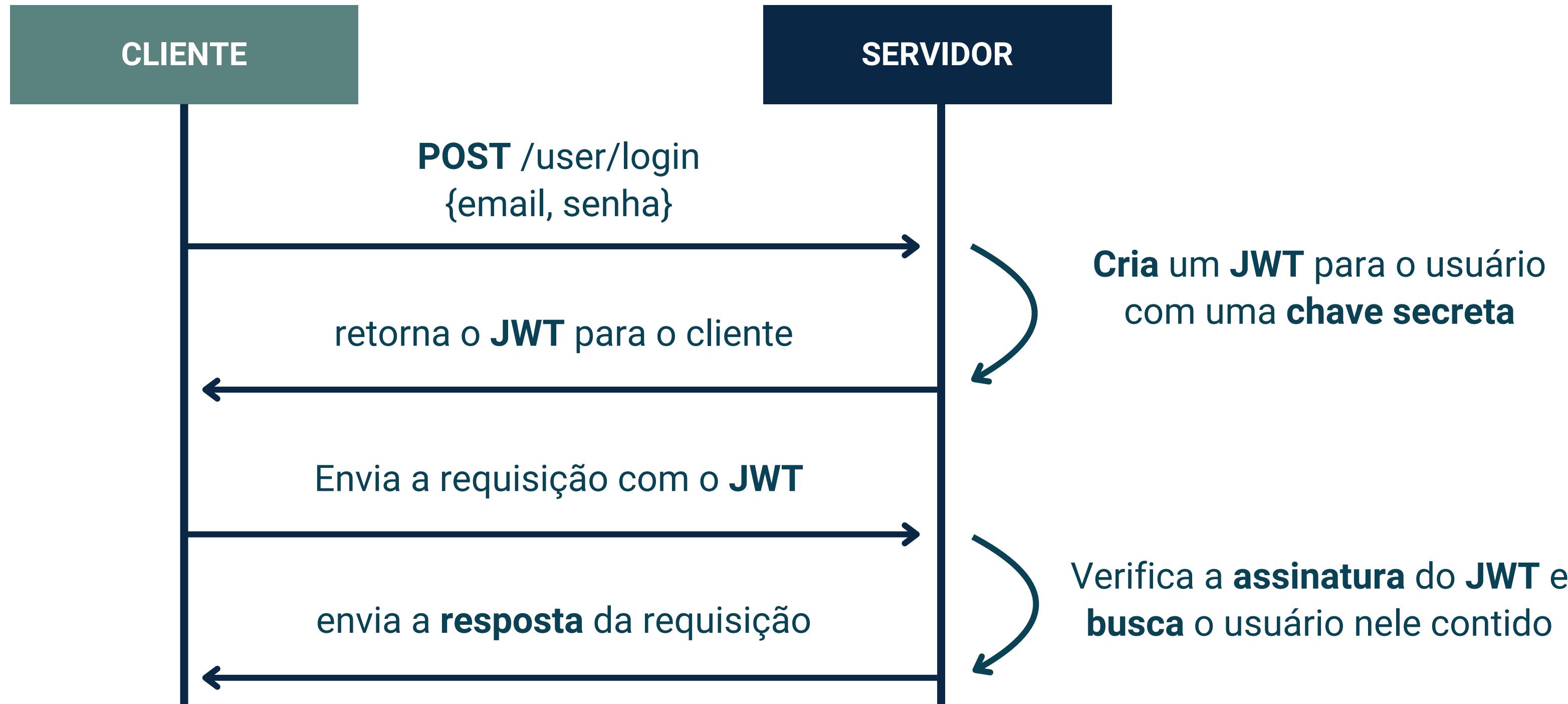


Cookies de Sessão  
ou JWT?

# Cookies



# JSON WebToken {...}



# Cookies de Sessão ou JWT?

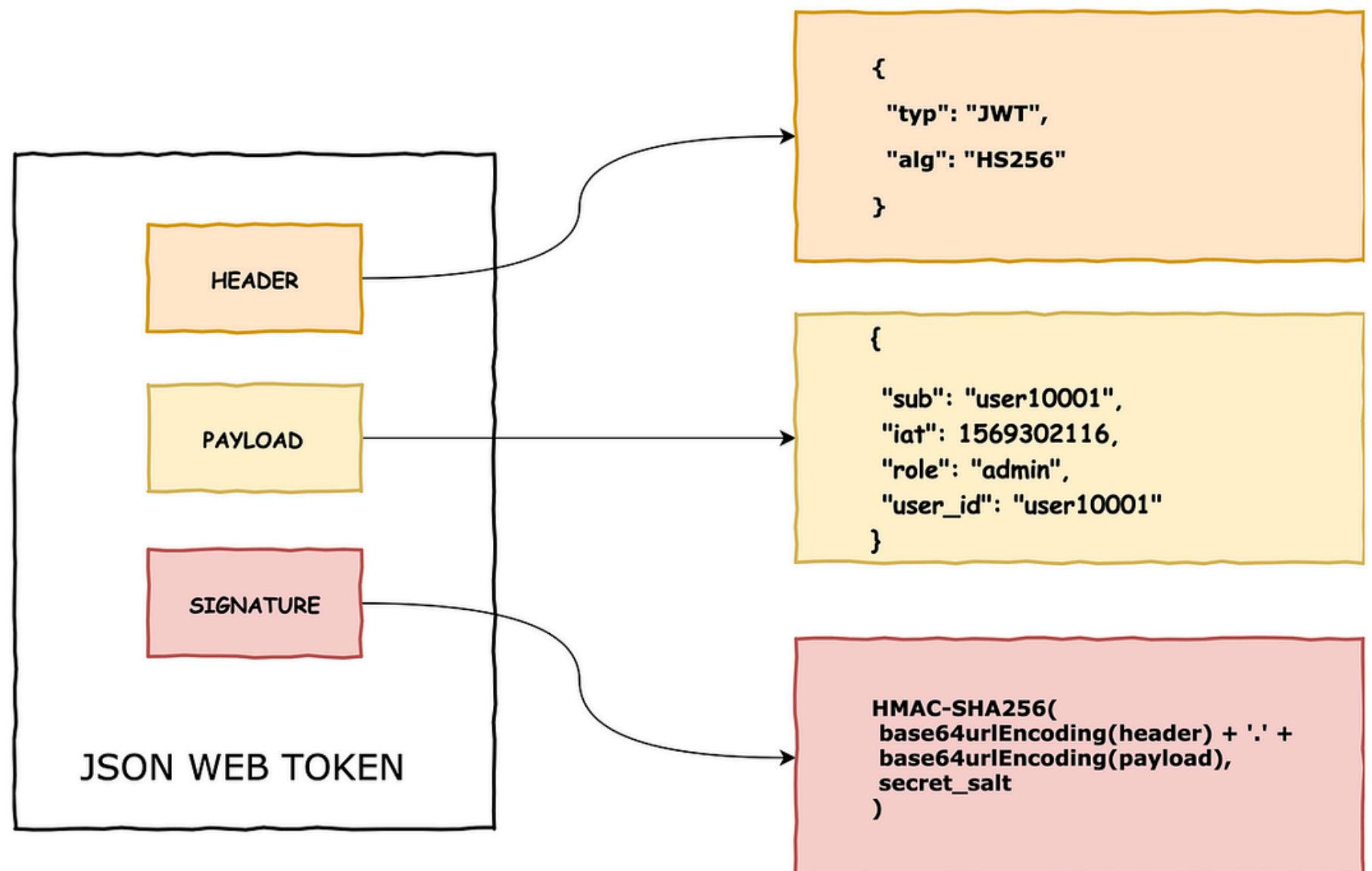
## Cookies de Sessão

- Armazenado no navegador como cookies e enviado automaticamente ao servidor em cada requisição;
- O servidor **mantém o estado da sessão**, o que não é ideal para **escalabilidade** em sistemas distribuídos.

## JWT

- Armazenado no **Local Storage** ou **Session Storage** do navegador e deve ser **incluído** nas requisições.
- O servidor **não guarda** as informações do usuário, apenas verifica ela.
- Permite verificação **sem dependência** de um estado no servidor, facilitando sistemas distribuídos.

# Estrutura do JWT



**Construído em 3 partes**

O **header**, o **payload** e **signature**. Cada seção é codificada em **base64** e separadas por **um ponto**.

# JWT em detalhes

```
{  
  "alg": "HS256",  
  "typ": "JWT"  
}
```

## Header

Consiste no tipo de token (JWT) e no algoritmo de assinatura usado. Ex: HMAC SHA256 ou RSA.

```
{  
  "iss":  
    "example_issuer",  
  "sub":  
    "user_id123",  
  "exp": 1644768000,  
  "iat": 1644744000  
}
```

## Payload

Contém as informações sobre o usuário e dados adicionais. Ex: Horário de emissão e expiração do token.

```
signature = HMAC-  
SHA256(base64urlEncode(header)  
+ "." +  
base64urlEncode(payload),  
secret_key)
```

## Signature

Verifica a integridade e autenticidade do token, criada usando uma chave secreta e um algoritmo de hash.

# JWT na prática

Token:

```
eyJhbGciOiJIUzUxMiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmcFtZSI6ImFsaWN1IiwiZXhwIjoxNzMwODE2NjY0LCJyb2x1IjoiYWRtaW4iLCJwZXJtaXNzaW9ucyI6WyJyZWFKIiwid3JpdGUiXX0.M12rvNkYVHqs6QNUPggu513Sb0mq8HbhRkdSii eYQjUiTnQ44a0aNnxXWw8Ak7j49pUhYjENeWqn-uGd-r5B-Q
```

Ponto de separação

Ponto de separação

# JWT na prática - Header e Payload

```
// Header
{
  "alg": "HS256",
  "typ": "JWT"
}

// Payload
{
  "sub": "1234567890",
  "name": "José Santos",
  "iat": 1516239022
}
```

## Convertendo para base64 e concatenando

```
header_base64 =
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9

payload_base64 =
eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4gRG9lIiwiaWF0IjoxNTE2MjM5MDIyfQ

string_para_assinar = header_base64 + "." +
payload_base64
```

# JWT na prática - Assinatura

```
secret_key = "caneta_azul"  
signature = HMAC_SHA256(secret_key, string_para_assinar)
```

## Convertendo para base64 e concatenando

header\_base64.payload\_base64.signature\_base64

## Obtendo o token

eyJhbGciOiJIUzUxMiIsInR5cCI6IkpXVCJ9.  
eyJ1c2VybmcFtZSI6ImFsawN1IiwidXhwIjoxNzMw0DE2NjY  
ØLCJyb2x1IjoiYWRtaW4iLCJwZXJtaXNzaW9ucyI6WyJyZWFKIiwid3JpdGUiXXØ.M12rvNkYVHqs6QNUPgu  
513Sb0mq8HbhRkdSiieYQjUiTnQ44a0aNcnxXWw8Ak7j49pUhYjENeWqn-uGd-r5B-Q

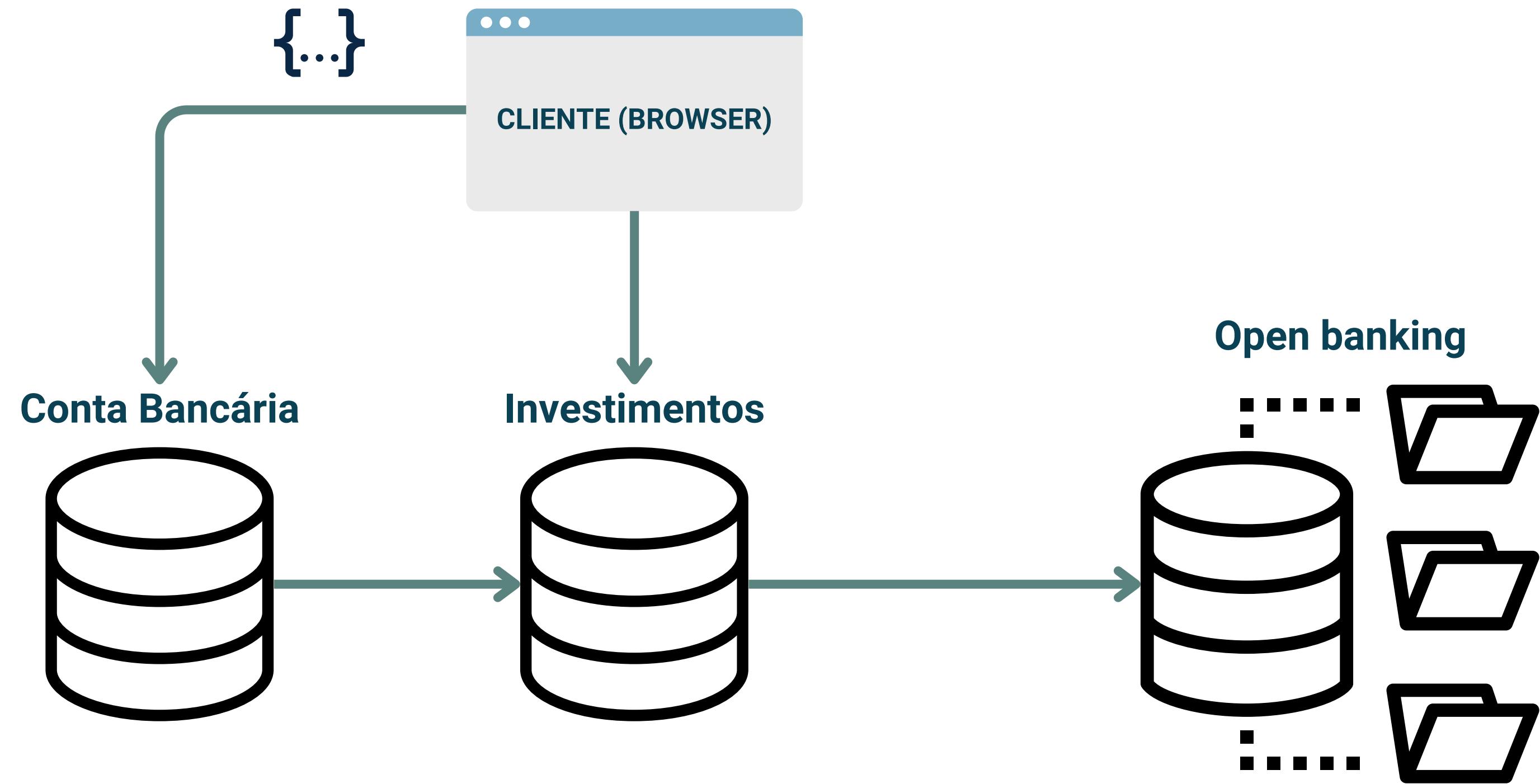
# JWT na prática - Validação

**A chave utilizada para assinar é conhecida apenas pelo servidor!**

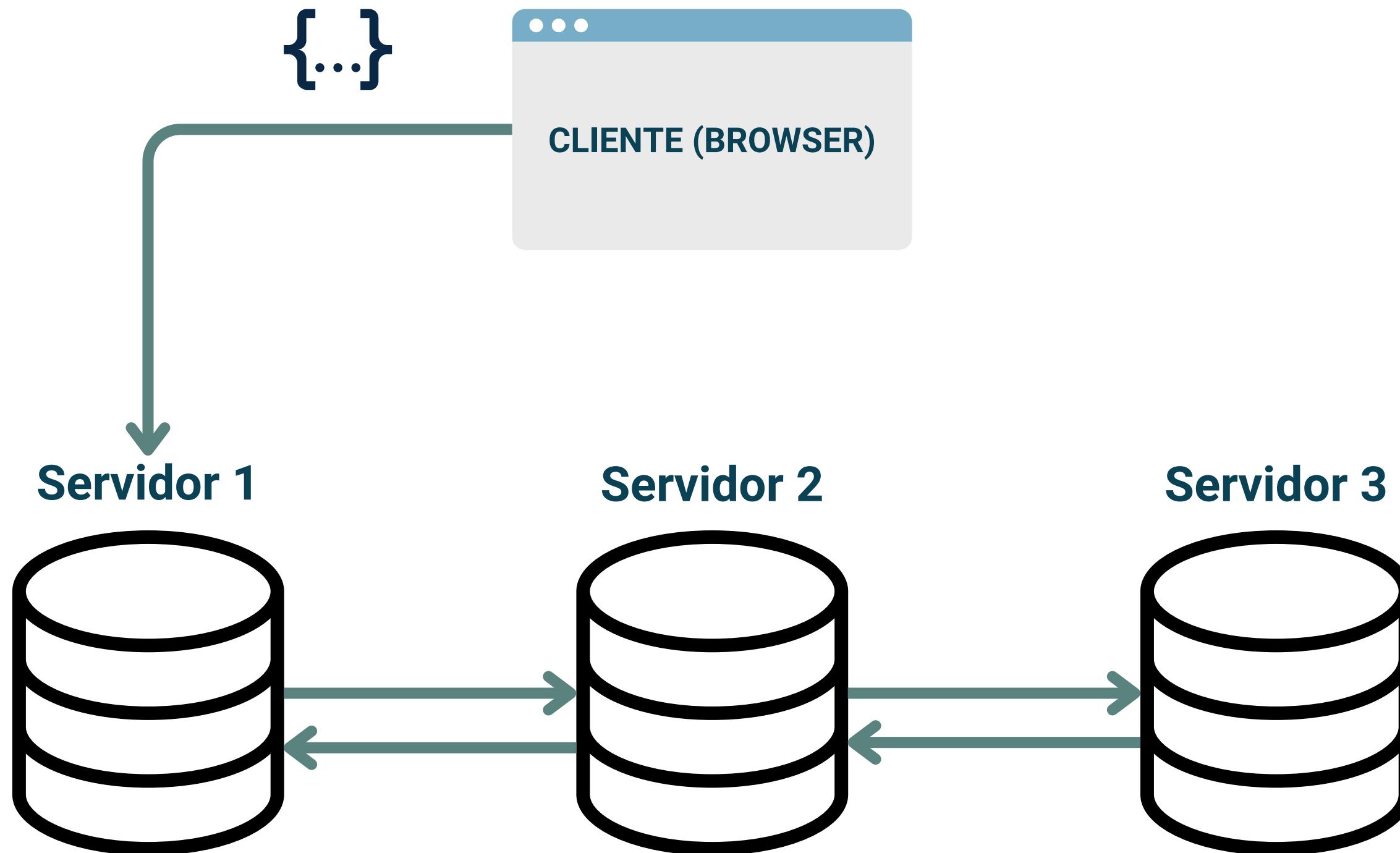
```
secret_key = "caneta_azul"
```

O servidor **recebe** a solicitação e **verifica a assinatura** usando a **chave secreta** que foi usada para assiná-lo. Se o JWT for válido, o servidor **extrai as informações** contidas nele e as **usa para determinar** quais ações o usuário está **autorizado** a executar.

# JWT na prática - Exemplos



# JWT na prática - Exemplos



# JWT vale a pena?

## Vantagens

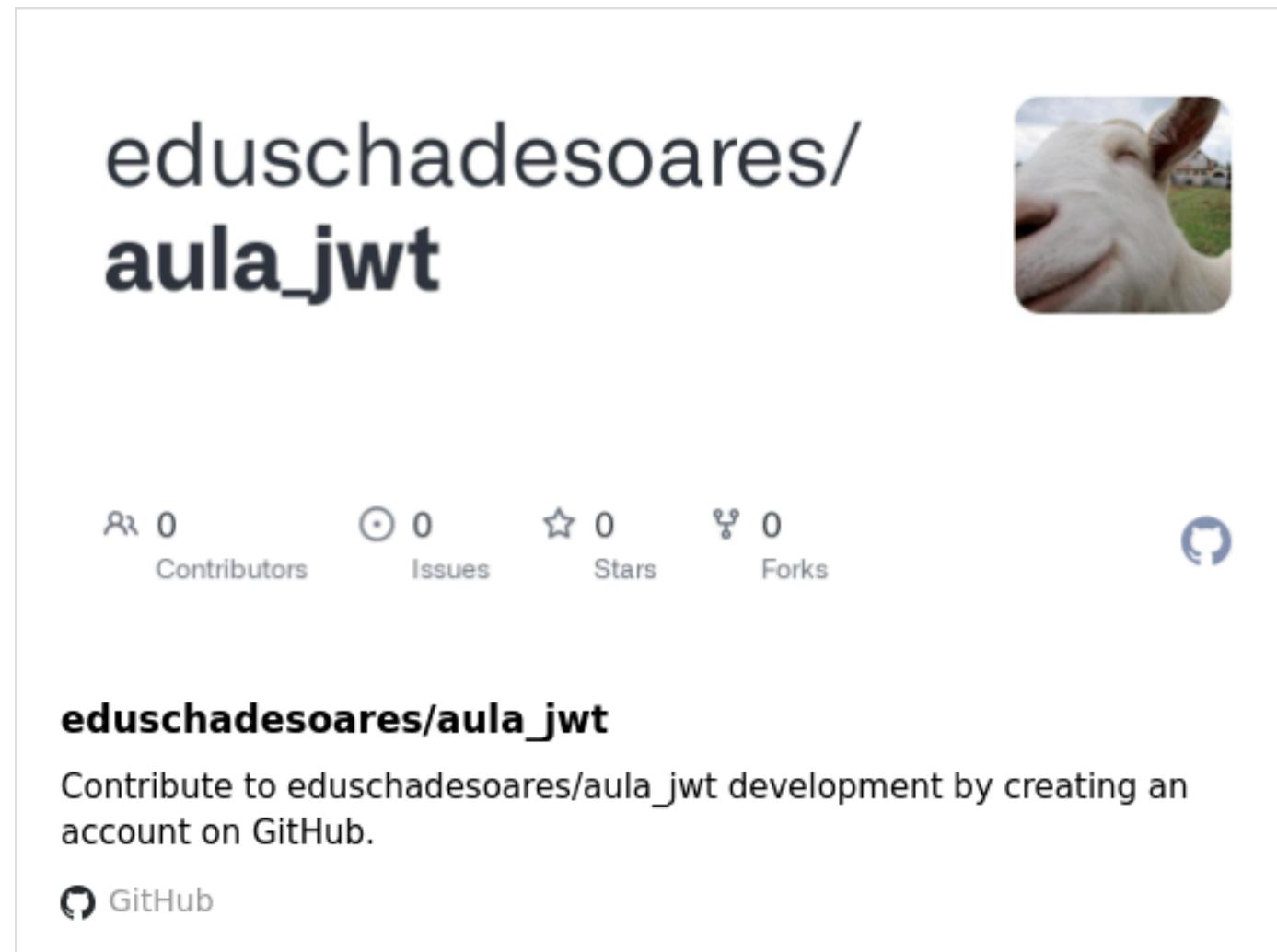
- **Portátil:** pode ser processado em várias plataformas, **web** e **mobile**.
- As bibliotecas de **JSON** são comuns na **maioria** das linguagens de programação.
- **Protegido contra adulteração** devido à **chave secreta** armazenada no lado do servidor.
- O servidor **não precisa armazenar** nenhuma informação de sessão, devido à natureza **sem estado** do token JWT.
- Uma vez que o JWT **contém informações** do usuário, os servidores **evitam consultar** o banco de dados para obter detalhes do usuário em **cada solicitação, melhorando** o desempenho.

## Desvantagens

- JWTs podem se carregar **dados extensos** de usuários, levando a um **aumento no tráfego de rede**.
- **Não se deve armazenar** informações **sigilosas** nele.
- Os JWTs permanecem válidos até **expirarem**. Revogar um JWT antes da expiração requer **complexidade adicional**, como usar uma **lista negra de tokens**.
- São normalmente **imutáveis** uma vez emitidos. Se a **função** ou as **permissões** de um usuário mudarem, precisará ser feito **login novamente** para obter um token atualizado.

# JWT na prática - Mão na massa!

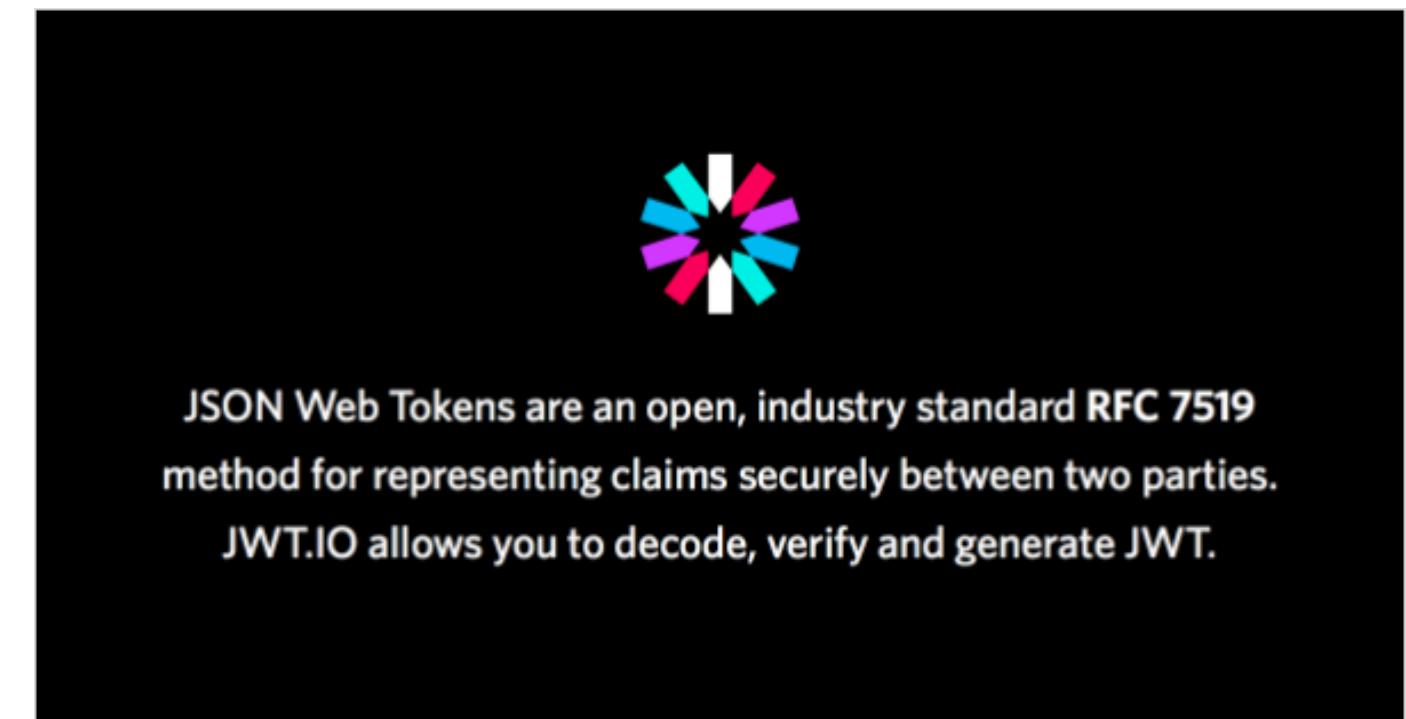
[github.com/eduschadesoares/aula\\_jwt](https://github.com/eduschadesoares/aula_jwt)



eduschadesoares/  
**aula\_jwt**

0 Contributors   0 Issues   0 Stars   0 Forks

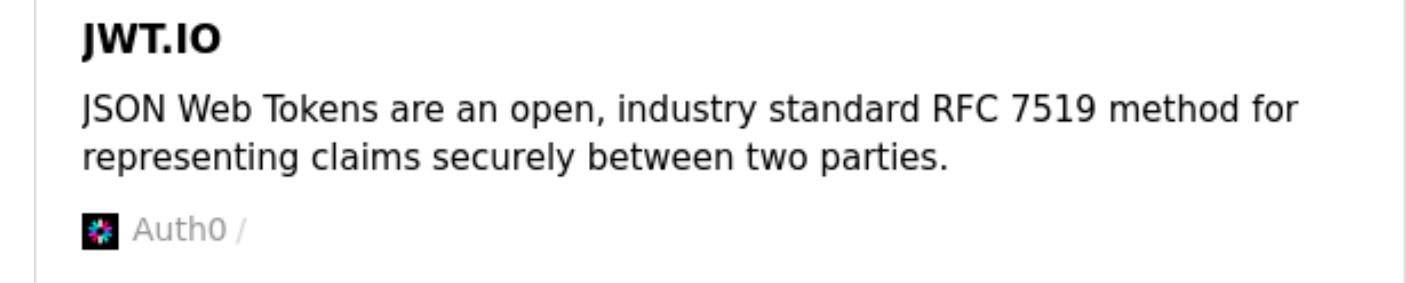
[GitHub](#)



**jwt.io**

JSON Web Tokens are an open, industry standard **RFC 7519** method for representing claims securely between two parties.

JWT.IO allows you to decode, verify and generate JWT.



**JWT.IO**

JSON Web Tokens are an open, industry standard RFC 7519 method for representing claims securely between two parties.

[Auth0 /](#)

# Referências

- G. MCGRAW, "Software security," in **IEEE Security & Privacy**, vol. 2, no. 2, pp. 80-83, March-April 2004, doi: 10.1109/MSECP.2004.1281254.
- SPILCA, Laurentiu. **Spring security in action**. Simon and Schuster, 2020.

## Consultas aos sites:

### **JWT:**

<https://medium.com/@codealfi/understanding-jwt-authentication-benefits-and-limitations-3c388dba172e>  
<https://www.miniorange.com/blog/what-is-jwt-json-web-token-how-does-jwt-authentication-work/>  
<https://dev.to/jaypmedia/jwt-explained-in-4-minutes-with-visuals-g3n>

### **JSON:**

<https://www.json.org/json-en.html>

### **Curiosidades:**

<https://jovemnerd.com.br/podcasts/nerdcast/nerdcast-380-espionagem-digital>

### **Lei na internet:**

[https://www.planalto.gov.br/ccivil\\_03/\\_ato2015-2018/2018/lei/l13709.htm](https://www.planalto.gov.br/ccivil_03/_ato2015-2018/2018/lei/l13709.htm)

<https://gdpr-info.eu/>

# Obrigado!

Perguntas sobre a aula?