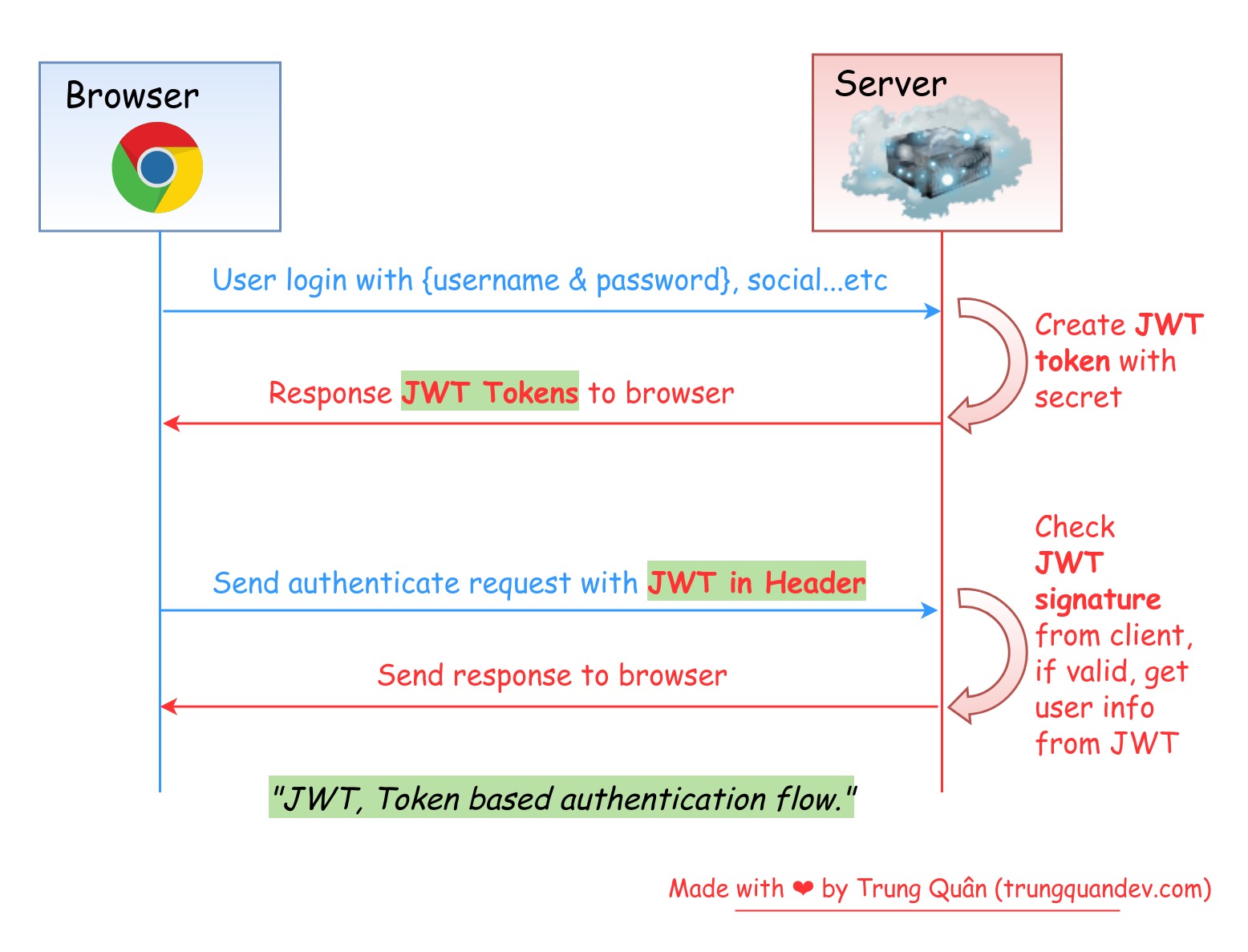
“JWT là một JSON object được định nghĩa trong chuẩn **[RFC 7519](https://tools.ietf.org/html/rfc7519" \t "/home/sonnguyen/Documents\\x/_blank)** như là một cách an toàn để trao đổi thông tin giữa hai bên. Và Token thì bao gồm một header, một payload và một chữ ký.”

1. **JWT hoạt động thế nào ?**



Server sẽ ký một mã JWT sau đó gửi về cho phía client lưu trữ, những request tiếp theo từ client gửi lên thì phải đính kèm mã JWT này (thông thường là ở header), server sẽ check mã này và gửi lại response thành công hoặc thất bại tương ứng ngược về client.

“Đối với việc lưu trữ JWT ở client, sẽ có vài trường hợp, nếu như là trình duyệt web thì JWT có thể lưu vào **[Local Storage](https://developer.mozilla.org/en-US/docs/Web/API/Web_Storage_API/Local_storage" \t "/home/sonnguyen/Documents\\x/_blank)**, IOS app thì sẽ là **[Keychain](https://developer.apple.com/documentation/security/keychain_services" \t "/home/sonnguyen/Documents\\x/_blank)** và Android app sẽ lưu vào **[SharedPreferences](https://developer.android.com/reference/android/content/SharedPreferences" \t "/home/sonnguyen/Documents\\x/_blank).**”

1. **Cách tạo ra một mã JWT**

Có 3 thành phần quan trọng của nó lần lượt là: **HEADER**, **PAYLOAD** và **SIGNATURE**

**HEADER**: Đây là nơi chứa cái thông tin mà được dùng để trả lời cho câu hỏi: “Mã JWT được tính toán như thế nào?”

Ví dụ cho một cái Header, nó là một đối tượng JSON giống như thế này:

**{**

"typ": "JWT"**,**

"alg": "HS256"

**}**

Trong cái Header ví dụ trên, thì “**typ**” (viết tắt của type) là kiểu **Token**, ở đây chính là **JWT**.

Còn “**alg**” (viết tắt của **algorithm**) là thuật toán băm tạo ra chữ ký cho Token, ở ví dụ trên **HS256**là thuật toán có tên **HMAC-SHA256**, một thuật toán băm sử dụng khóa bí mật (**Secret Key**) để tính toán tạo ra chữ ký.

**HEADER**: Đây là nơi chứa những đữ liệu mà chúng ta muốn lưu lại trong JWT

**{**

"userId": "7j79y-kdjr8n4h-5jd8-5k39-cfk8ghr9wu"**,**

"username": "trungquandev17"**,**

"occupation": "Full stack web developer"**,**

// standard fields

"iss": "Trung Quan, author of blog: https://trungquandev.com"**,**

"iat": 1568456819**,**

"exp": 1568460419

**}**

Standard Fields, chúng nó là những trường tiêu chuẩn, là là optional nghĩa là các bạn có thể tạo hoặc không, nhưng mà nên tạo nhé vì nó sẽ hữu ích.

– “**iss**” viết tắt của Issuer là thông tin người tạo ra Token (không phải user đâu nhé, mà nó chính là tên cái hệ thống backend của các bạn chẳng hạn)

– “**iat**” viết tắt của Issued at, là nhãn thời gian lúc mà cái token được tạo.

– “**exp**” viết tắt của Expiration time, xác định thời gian hết hạn của Token

Ngoài ra còn nhiều standard fields nữa, các bạn hãy tham khảo ở đây nhé, thấy cái nào phù hợp thì dùng cho ứng dụng của các bạn.  
*[https://en.wikipedia.org/wiki/JSON\_Web\_Token#Standard\_fields](https://en.wikipedia.org/wiki/JSON_Web_Token" \l "Standard_fields" \t "/home/sonnguyen/Documents\\x/_blank)*

**SIGNNATURE:** Chữ ký

// signature algorithm

const data = base64urlEncode**(**header**)** + “.” + base64urlEncode**(**payload**)**;

const hashedData = Hash**(**data, secret**)**;

const signature = base64urlEncode**(**hashedData**)**;

Đầu tiên, chúng ta sẽ Encode (chuyển đổi) 2 cái **Header**và **Playload**ở trên theo kiểu **[Base64URL](https://kjur.github.io/jsjws/tool_b64uenc.html" \t "/home/sonnguyen/Documents\\x/_blank)** [Encoder](https://kjur.github.io/jsjws/tool_b64uenc.html" \t "/home/sonnguyen/Documents\\x/_blank), và nối 2 chuỗi nhận được lại (cách nhau bởi dấu chấm “.”) rồi gán nó vào một biến là data.

Tiếp theo sẽ **Hash** (băm) cái data đó bằng “**alg**”, chính thuật toán tạo chữ ký mà chúng ta đã định nghĩa ở trên Header (HS256 – HMAC-SHA256) kèm với một chuỗi bí mật **secret**(chuỗi secret này sẽ được đặt tùy vào lập trình viên của mỗi dự án và đảm bảo không được để lộ chuỗi này ra ngoài, có thể đưa vào biến môi trường **ENV**.)

Sau khi băm xong ở trên thì thực hiện Encode tiếp một lần nữa cái dữ liệu băm đó dưới dạng Base64URL Encode, và chúng ta sẽ thu được chữ ký “Signature”.

const headerEncode = base64urlEncode**(**header**)**; // ví dụ kết quả: eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9

const payloadEncode = base64urlEncode**(**payload**)**; // ví dụ kết quả: eyJ1c2VySWQiOiJiMDhmODZhZi0zNWRhLTQ4ZjItOGZhYi1jZWYzOTA0NjYwYmQifQ

const data = headerEncode + "." + payloadEncode;

const hashedData = Hash**(**data, secret**)**;

const signature = base64urlEncode**(**hashedData**)**; // ví dụ kết quả: xN\_h82PHVTCMA9vdoHrcZxH-x5mb11y1537t3rGzcM

// cuối cùng thì mã JWT theo đúng cấu trúc header.payload.signature sẽ trông như sau:

const JWT = headerEncode + "." + payloadEncode + "." + signature;

// Kết quả:

"eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1c2VySWQiOiJiMDhmODZhZi0zNWRhLTQ4ZjItOGZhYi1jZWYzOTA0NjYwYmQifQ.xN\_h82PHVTCMA9vdoHrcZxH-x5mb11y1537t3rGzcM"