JWT (JSON Web Token)

JWT stands for JSON Web Token. In simple terms, it's like a digital passport that securely carries information between parties. It's commonly used for authentication and authorization in web development.

This information can be verified and trusted because it is digitally signed. JWTs can be signed using a secret (with the **HMAC** algorithm) or a public/private key pair using **RSA** or **ECDSA**.

## When should you use JSON Web Tokens?

**Authorization**: This is the most common scenario for using JWT. Once the user is logged in, each subsequent request will include the JWT, allowing the user to access routes, services, and resources that are permitted with that token. Single Sign On is a feature that widely uses JWT nowadays, because of its small overhead and its ability to be easily used across different domains.

**Information Exchange**: JSON Web Tokens are a good way of securely transmitting information between parties. Because JWTs can be signed—for example, using public/private key pairs—you can be sure the senders are who they say they are. Additionally, as the signature is calculated using the header and the payload, you can also verify that the content hasn’t been tampered with

## JSON Web Token structure

* **Header:** This part typically consists of two parts: the type of the token (JWT) and the signing algorithm being used, such as HMAC SHA256 or RSA.
* **Payload:** This is where the actual claims (information) are stored. It contains information like user ID, expiration time, or any custom data.
* **Signature:** To make sure the message wasn't changed along the way, a signature is created using the encoded header, the encoded payload, a secret key, and the algorithm specified in the header.

These three parts are then concatenated together with dots between them, forming a JWT like this: ‘header.payload.signature’. This structure allows the JWT to be easily transmitted and verified.