JWT 🎉

miguel.garrido@ironhack.com

JWT

JSON Web Token, also know as JWT, is an **open standard that provides a simple and self-contained format for securely sending data between parties.**

JWT

JWT is a compact, secure way to transmit information between parties as a JSON object.

It is commonly used for **authentication** and **authorisation** in web applications.

JWT

1. When a user logs in, the server generates a JWT (a signed token) and sends it back to the client.

2. The client stores this token (usually in localStorage or cookies) and sends it with future requests.

3. The server verifies the token to ensure the user is authenticated before granting access.

JWT

Stateless

The server doesn’t need to store session data (unlike traditional sessions).

Secure

Tokens are digitally signed (with a secret/key) to prevent tampering.

Portable

Can be used across different services (APIs, microservices).

Structure of JWT

A JWT is structured using **Base64Url** encoding for its header and payload, but its security comes from a cryptographic signature (e.g., HMAC-SHA256 for HS256).

The signature prevents tampering, while Base64Url ensures URL-safe transmission

Structure of JWT

Header

Contains token type & signing algorithm (e.g., HS256).

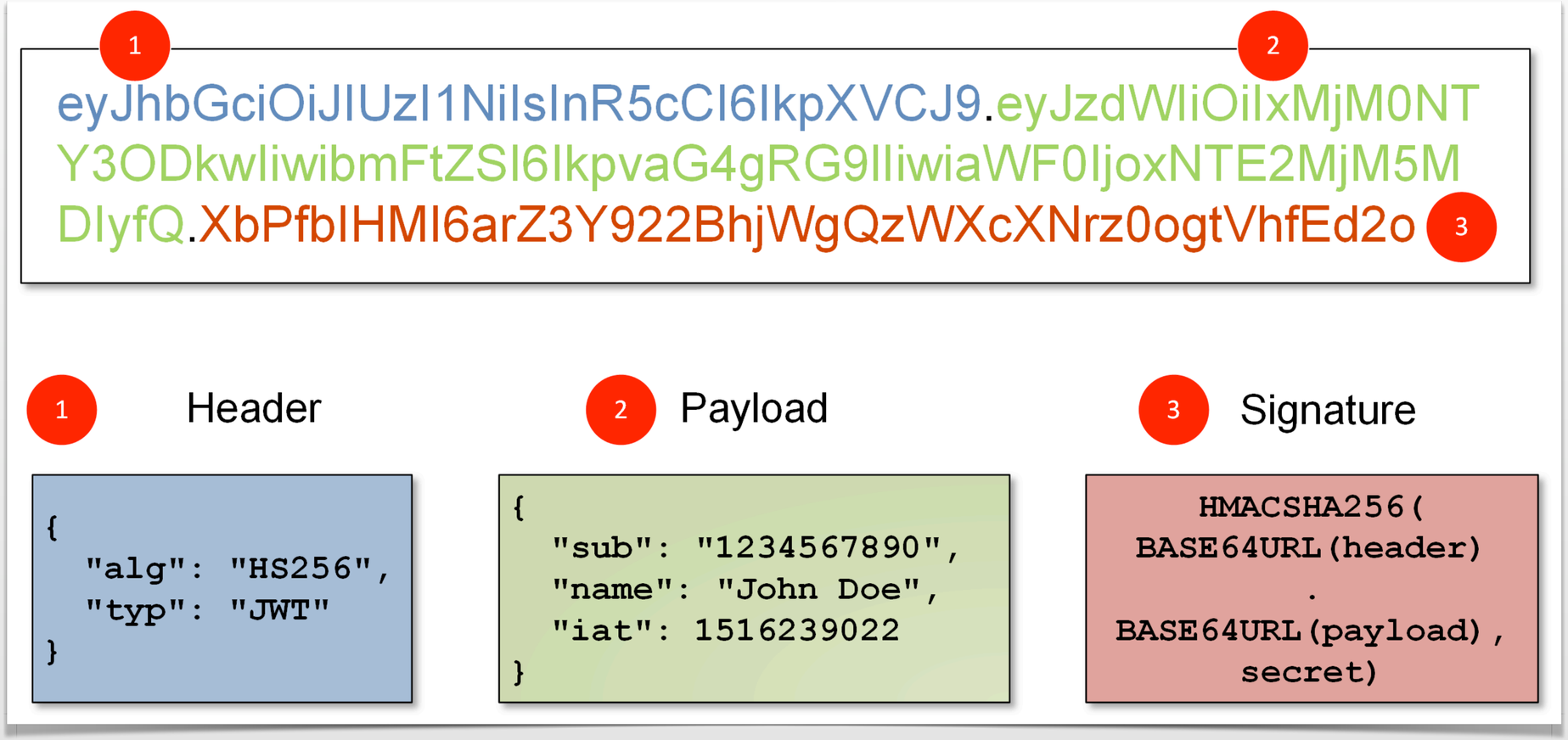
Payload

Contains claims (user data, expiration time, etc.).

Signature

Combination of **Header**, **Payload** and a **Secret Key**.

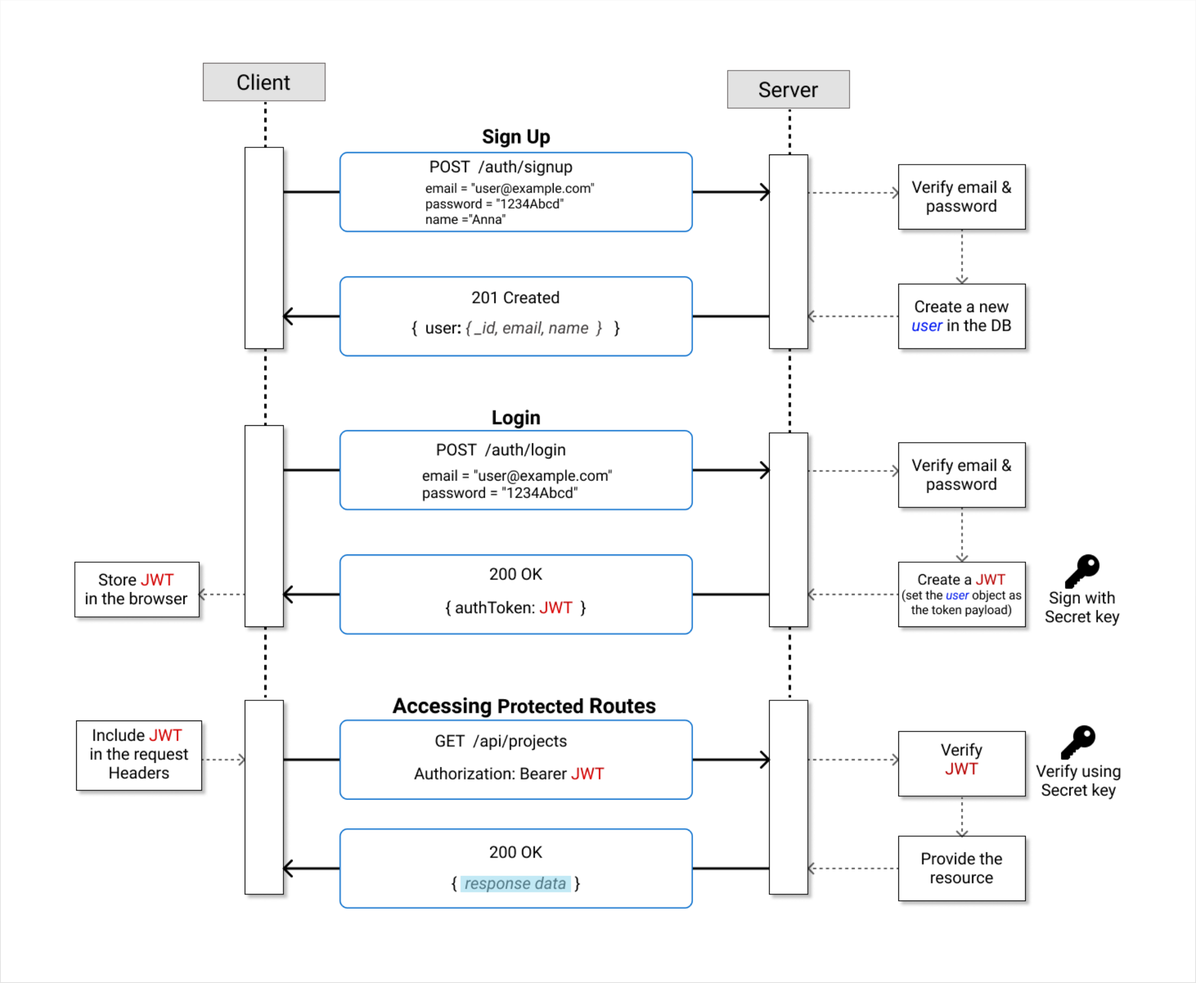
Structure of JWT



Authentication Flow

Method	URL	Request Headers	Request Body
POST	`/auth/signup`	–	{ email, password, name }
POST	`/auth/login`	–	{ email, password }
GET	`/auth/verify`	Authorization: Bearer < JWT >	–

Authentication Flow



IRON HACK

Recommendations

1. You always verify the signature on the server (e.g., using jsonwebtoken.verify()).
2. You never store sensitive data (passwords, credit cards) in the payload.
3. You use HTTPS to prevent token theft in transit.
4. You set short expiration times (e.g., exp: 1h) to limit risk if a token is stolen.

Why?

The signature ensures the token wasn’t tampered with.

Attackers can’t modify the payload without the secret/key.

Key takeaways

Signing ≠ Encryption

The signature is a cryptographic hash, not encrypted data. It proves integrity (payload wasn’t altered) but doesn’t hide the payload.

Payload is readable

Anyone can decode the Base64Url **header/payload**, but they can’t modify it without the secret key.