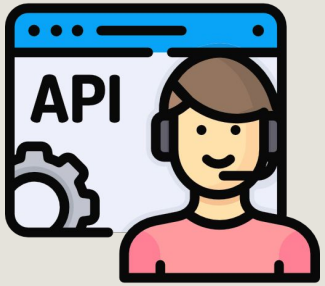
Web services (REST)

Web Development with Java

Agenda

Web Services

HTTP



REST & SOAP

Authentication


Web Services

Internet vs Web

Internet

Web

- global network of computers
- network protocols
- ≠ access ways
- no ownership

- way to share information
- **HTTP** - info travel
- **web browser** - access
- **URL** - navigation

Web 1.0

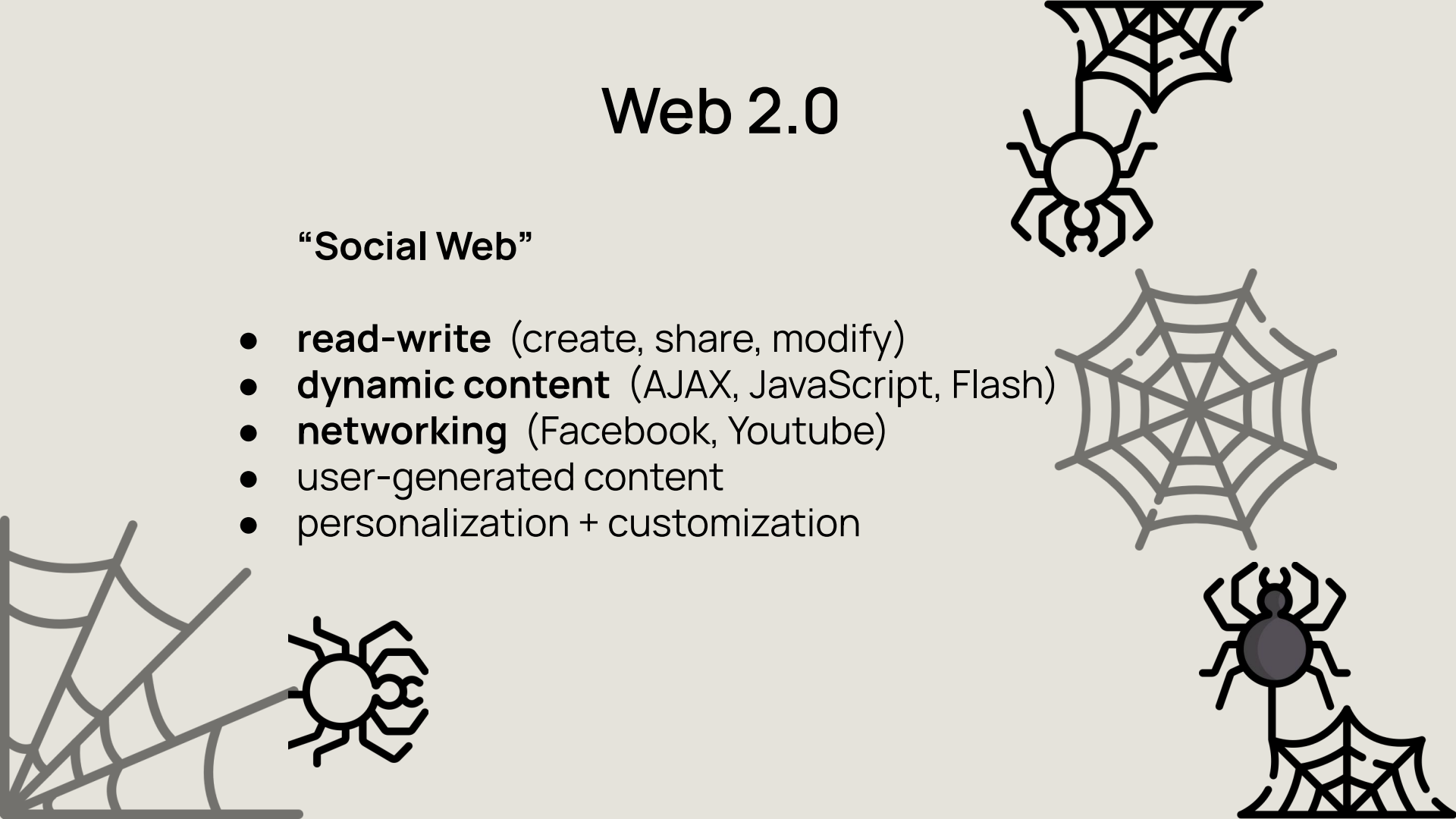
“Static Web”

- **read only** (one way communication)
- **static content** (HTML code changes)
- limited user interaction
- no user-generated content

Web 2.0

“Social Web”

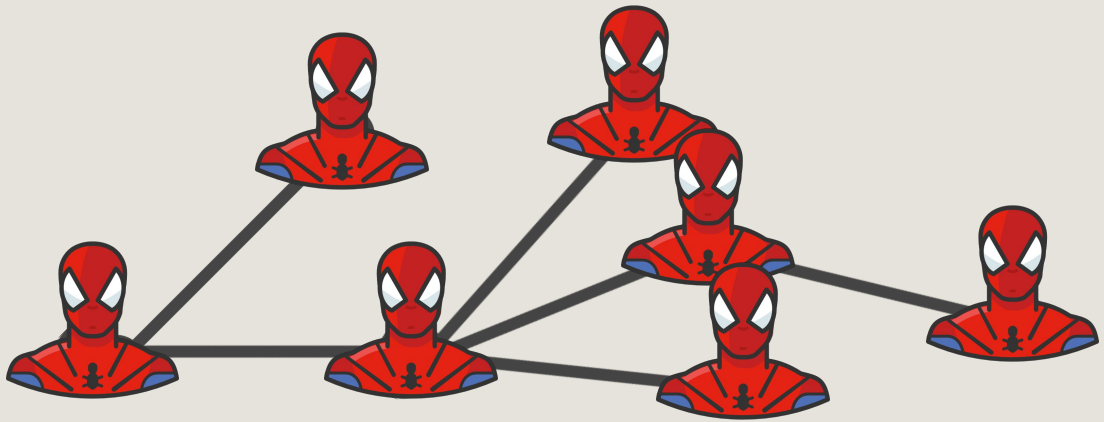
- **read-write** (create, share, modify)
- **dynamic content** (AJAX, JavaScript, Flash)
- **networking** (Facebook, Youtube)
- user-generated content
- personalization + customization



Web 3.0

“Semantic Web”

- **decentralization** (blockchain, smart contracts)
- **better privacy & security** (decentralized storage)
- **interoperability** (≠ blockchain networks)
- data ownership
- AI integration

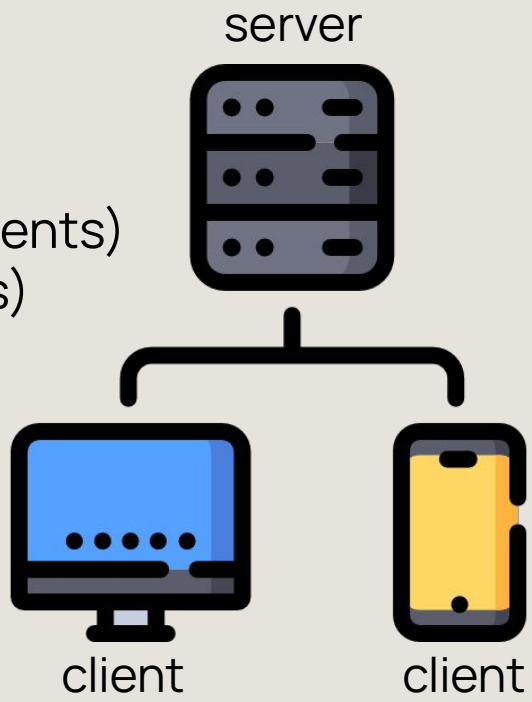



HTTP

HTTP

(Hypertext Transfer Protocol)

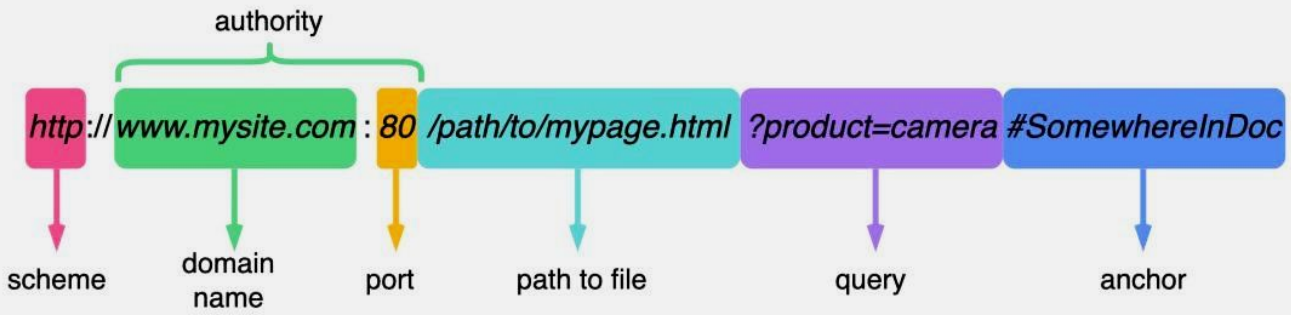
- **resource fetching** (HTML documents)
- **stateless** (independent requests)
- **anonymous**
- **request-response**



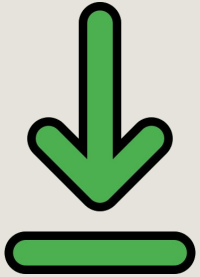
URL

URL

identify and locate resources

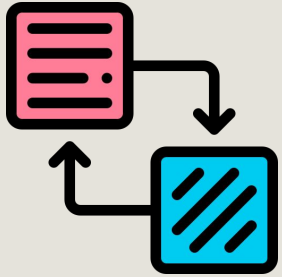


HTTP methods



GET
obtain data

POST
create resource



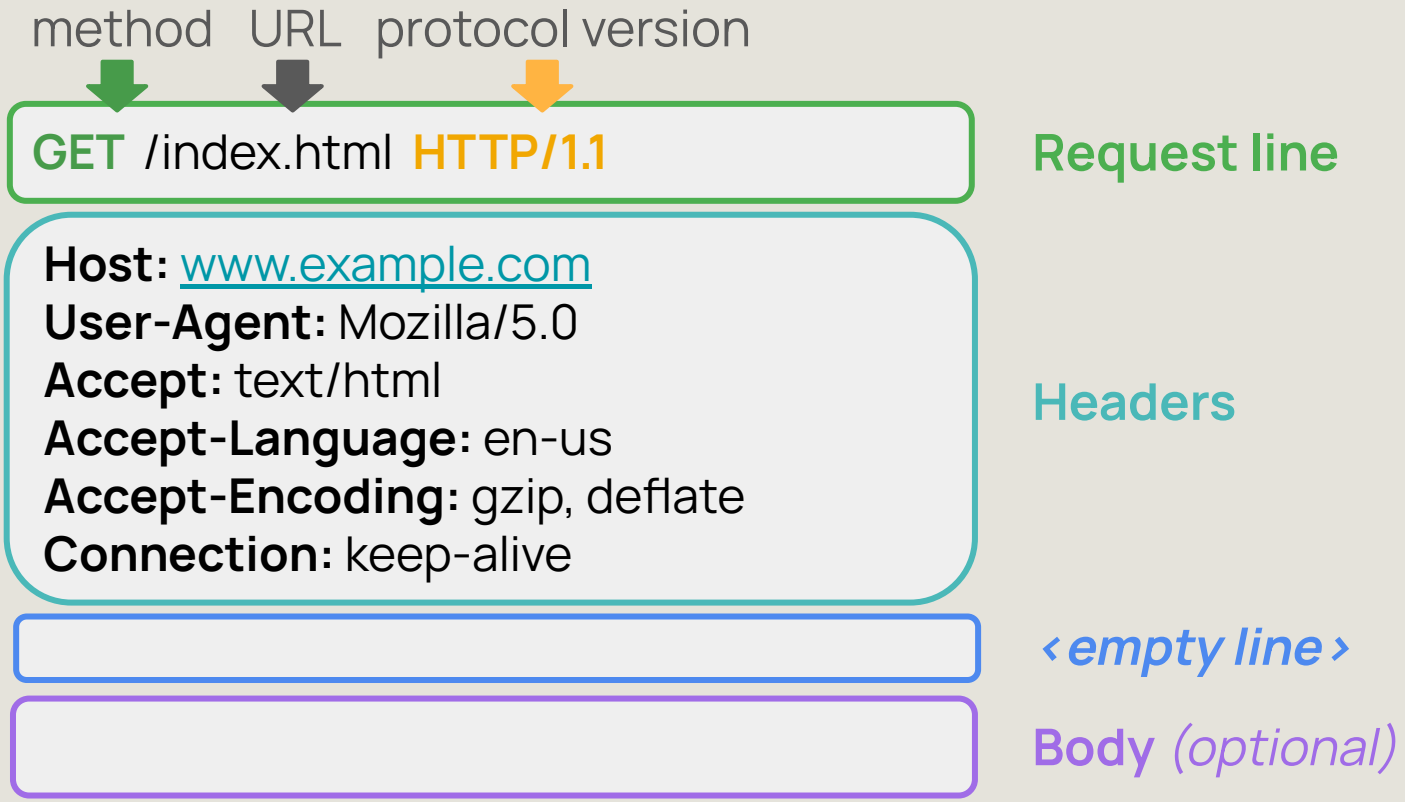
PUT
create/replace resource

PATCH
partial update

DELETE
destroy resource

idempotency

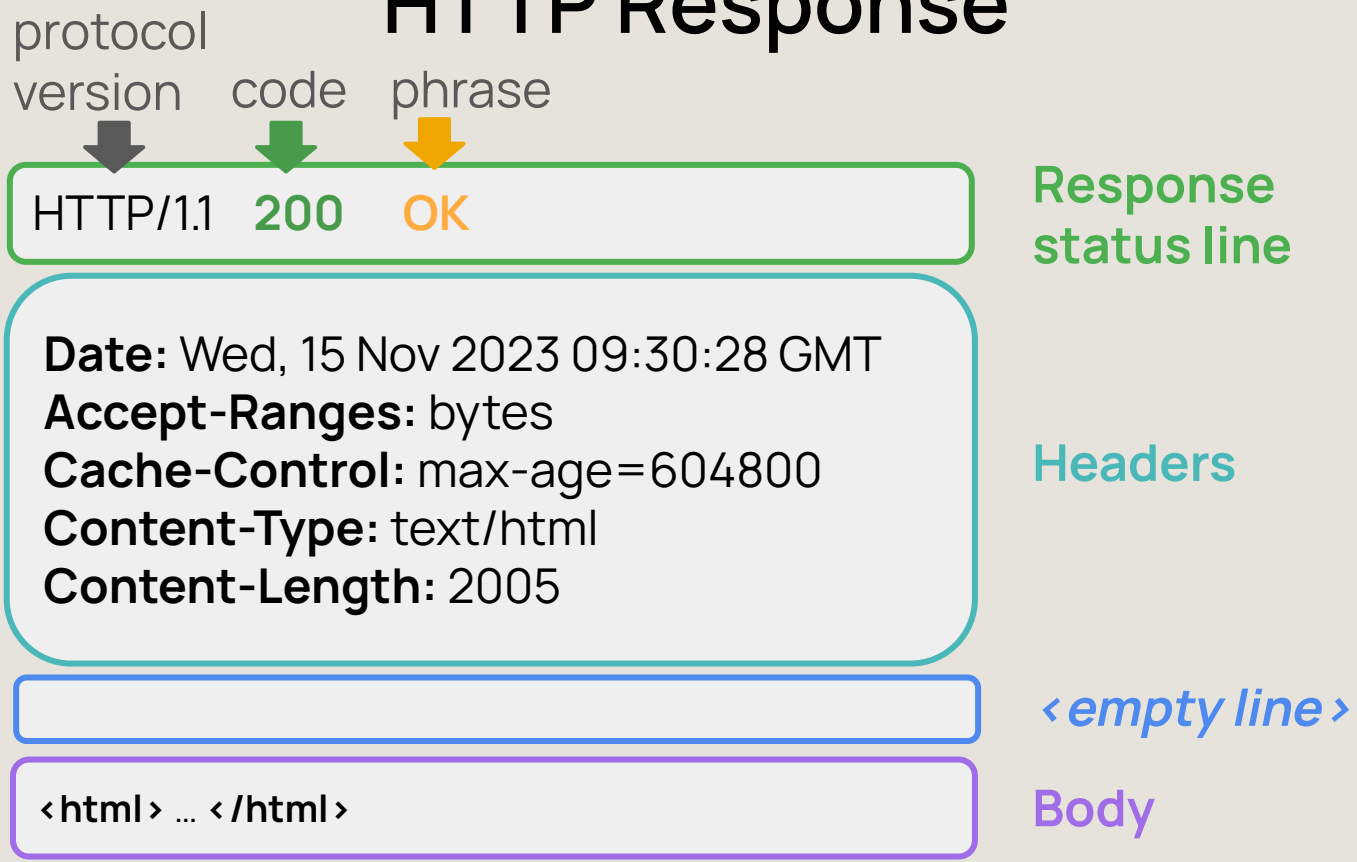
HTTP Request



JSON


{
"name": "Anna",
"age": 30,
"isStudent": false,
"address": {
"street": "123 Main St",
"city": "New York",
"postalCode": "10001"
},
"hobbies": ["travelling", "photography", "cooking"],
"hasPets": null
}

HTTP Response

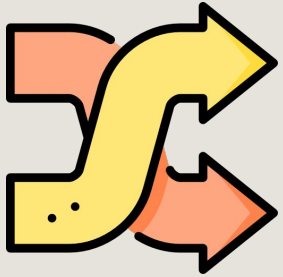


Status codes

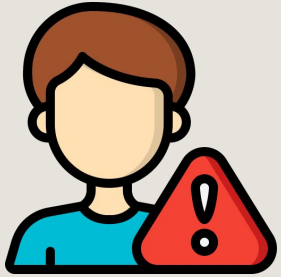
1xx
informational

2xx
success

200: OK
201: Created
202: Accepted

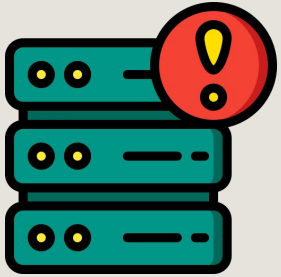


3xx
redirect



4xx
client error

400: Bad Request
401: Unauthorized
403: Forbidden
404: Not Found



5xx
server error

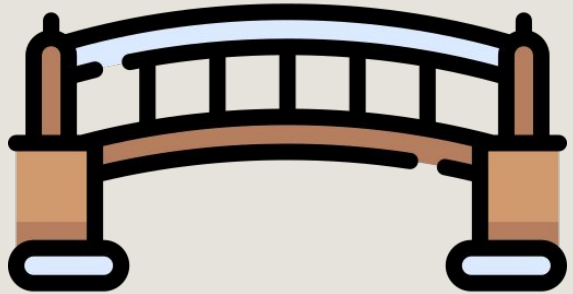
500: Internal Server Error
502: Bad Gateway

{ REST & SOAP }

REST

(Representational State Transfer)

- **architectural style**
- **standardized**
- **stateless**
- **cacheable** (response ♻)
- **client-server**

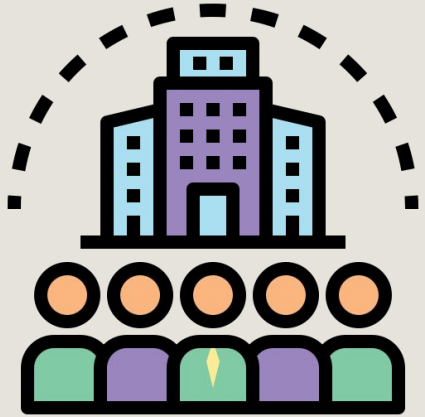


API = communication “bridge”

SOAP

(Simple Object Access Protocol)

- **protocol**
- **strict standards**
- **stateful**
- **built-in security**
- **flexible transport** (HTTP, TCP..)



enterprise-level

REST vs SOAP

flexible (JSON, XML, …)	**data & compliance**	strict (XML)
HTTPS	**security**	built-in
resource-oriented	**model**	function-driven
scalable web APIs	**use cases**	enterprise
faster, leaner	**performance**	slower

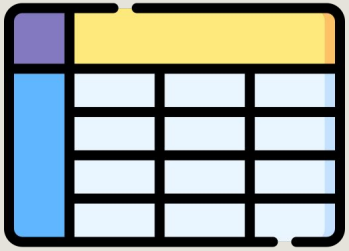

Authentication

Authentication vs Authorization

identity
(Who are you?)

rights
(What can you do?)

Encoding, Encryption, Hashing



- **public scheme**
- **reversible**
- (Base64, ASCII)

- **secret key**
- **reversible**
- (AES, RSA)

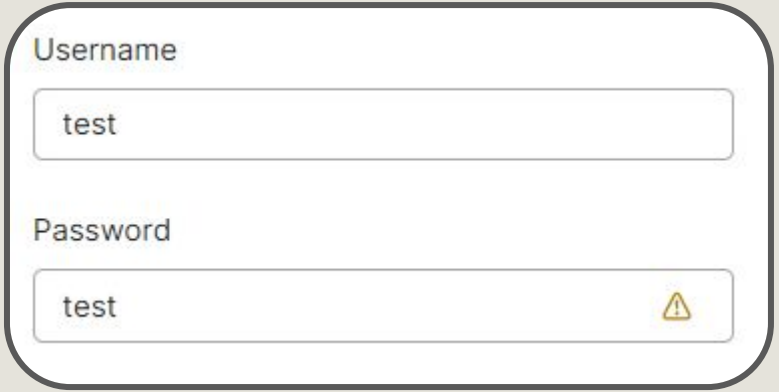
- **# algorithm**
- **irreversible**
- (SHA-256)

API Keys

- **simple identifier**
- **restrict** (within API)
- **header**
- **control traffic**

Basic Authentication

- **simple & easy**
- **authorization header**
- **format** (Base64)
- **limited** (reversible)



Basic *username:password*

dGVzdDp0ZXN0 = test:test

Bearer Authentication

- **secure & scalable**
- **token = access**
- **JWT** (JSON Web Token)
- **expiration time**

HTTPS!

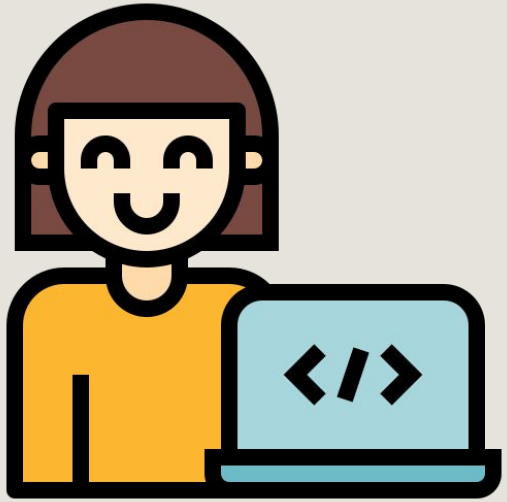
Bearer *‹token›*

JWT

- **compact & URL-safe**
- **expiration time**
- **SSO** (Single Sign-On)
  + API authentication

header (type + algorithm)
payload (claims)
signature (secret key)

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.ey
JzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6Ikpva
G4gRG9lIiwiaWF0IjoxNTE2MjM5MDIyfQ.SflKx
wRJSMeKKF2QT4fwpMeJf36POk6yJV_adQssw5c
```



Thank you!

Radina Nuneva

Resources

Cool links

HTTP methods difference
HTTP cats
JWT converter

Used resources:

Icons
Internet vs Web
Web versions
URL picture
JSON image