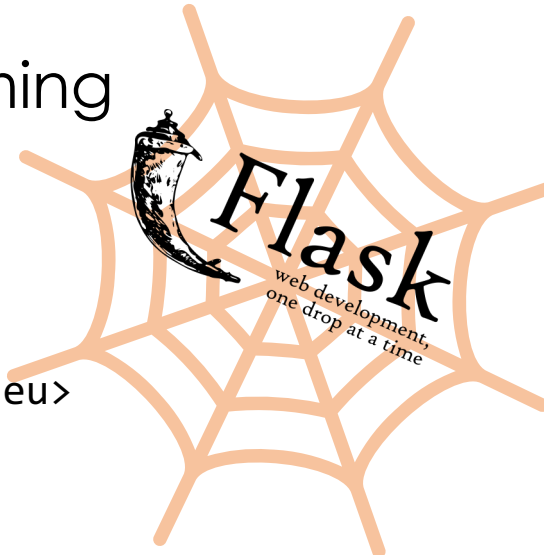
Web / BD

Introduction to Web Programming

2024-2025



Gérald Oster <gerald.oster@telecomnancy.eu>

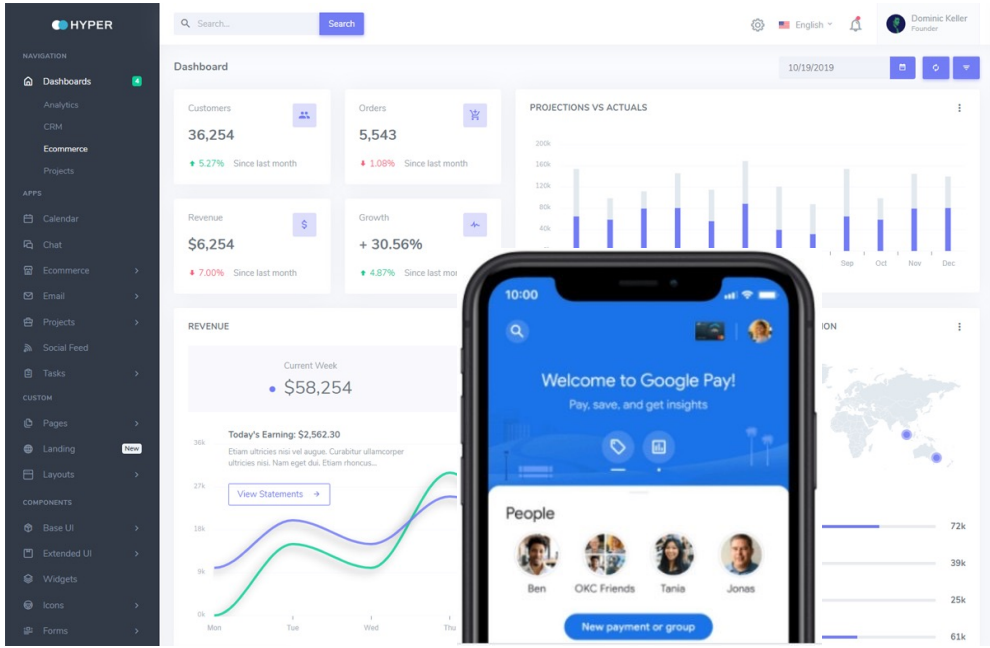
Organization of this part

- **4 x 1 hour - Lectures**
- **3 x 2 hours - Labs**
  - **Front-end: HTML / CSS**
  - **Back-end: Python/Flask/Database**
- **1 x 2 hours – Evaluation Lab (including some Database)**

Motivations

```
~/s/g/o/oclif  master  oclif --help
oclif: create your own CLI

VERSION
  oclif/1.12.1 darwin-x64 node-v10.11.0

USAGE
  $ oclif [COMMAND]

COMMANDS
  command  add a command to an existing CLI or plugin
  help     display help for oclif
  hook     add a hook to an existing CLI or plugin
  multi    generate a new multi-command CLI
  plugin   create a new CLI plugin
  single   generate a new single-command CLI

~/s/g/o/oclif  master
```

Command line interface (CLI)



(nowadays) Applications

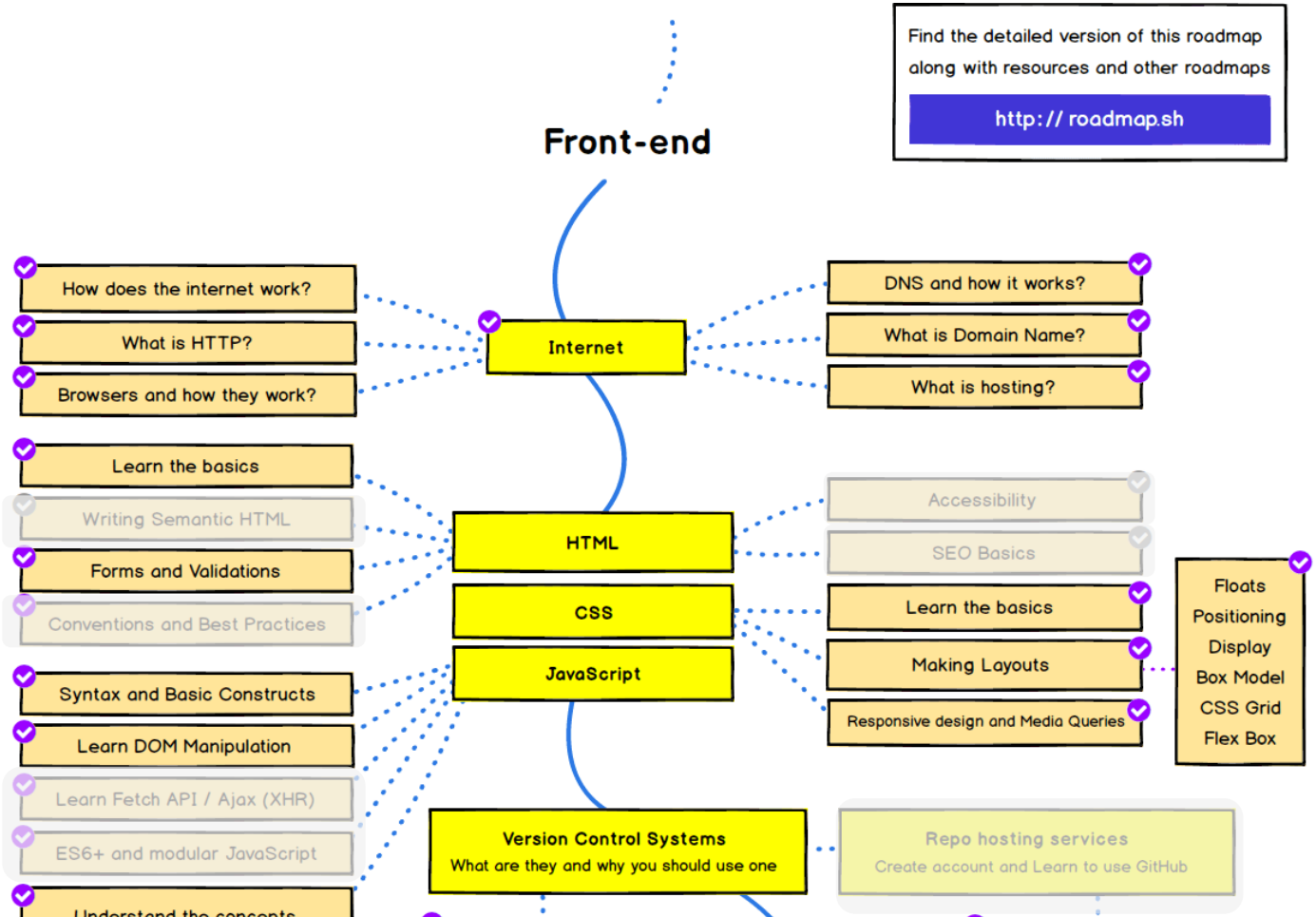
Disclaimer

- **NOT** a detailed course about Web Programming
- Demystification of Internet/Web
- Learn the concepts and the basics of web programming

This is just an introduction…

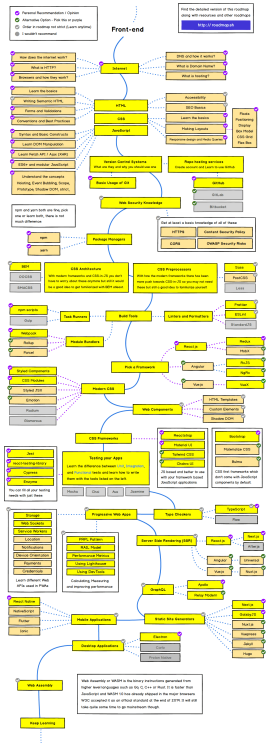
*… you will have to **learn by yourself at lot more***

[https://roadmap.sh/frontend]



[https://roadmap.sh/frontend]

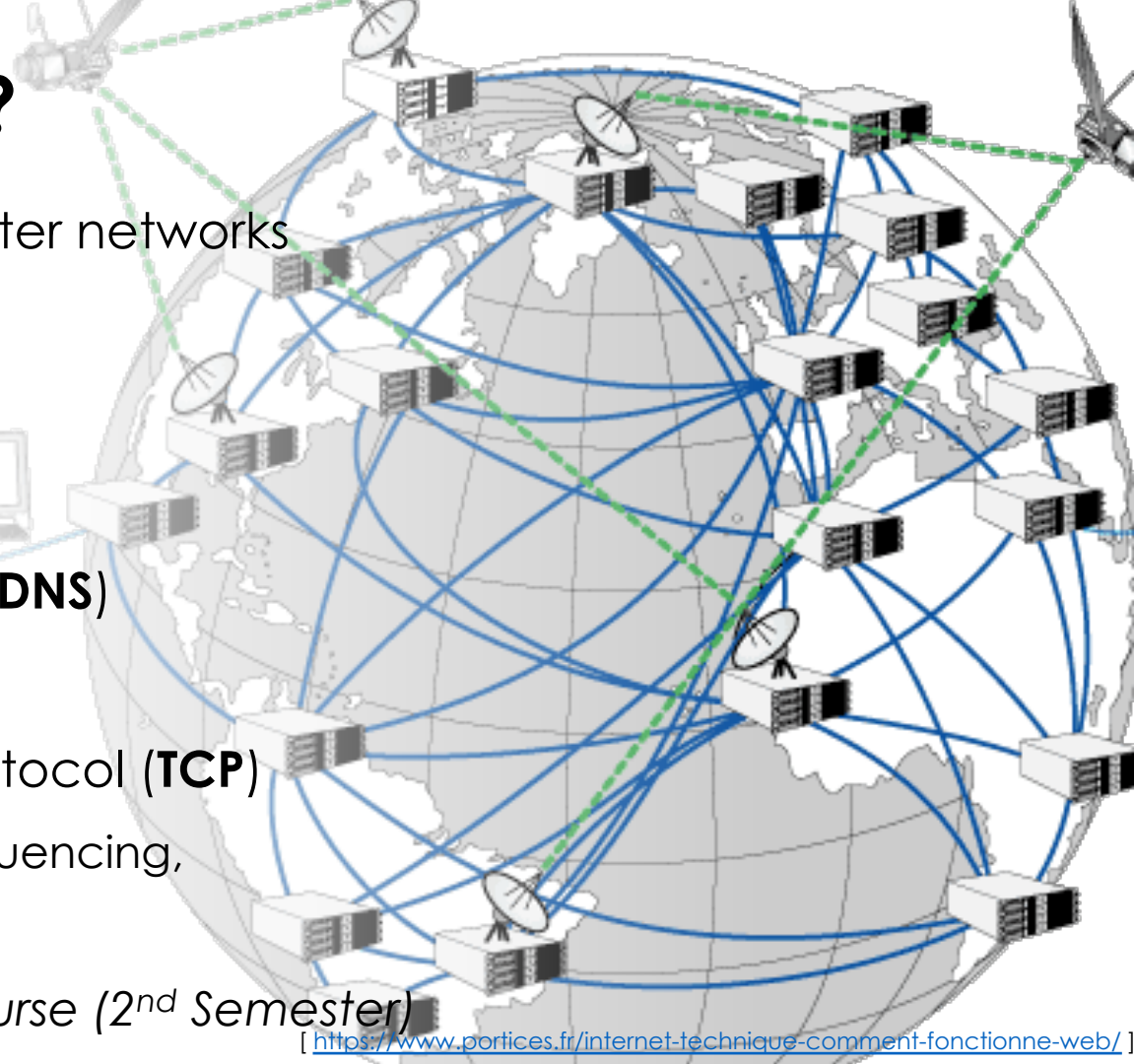
[https://roadmap.sh/backend]



What is Internet?

- Interconnected computer networks
  - Clients/Servers
- Internet Protocol (**IP**)
  - Routing
- Domain Name System (**DNS**)
  - Naming (IP <-> FQDNs)
- Transmission Control Protocol (**TCP**)
  - Packets: chunking, sequencing, resending, etc.

MORE *details in the Network Course (2nd Semester)*



[https://www.portices.fr/internet-technique-comment-fonctionne-web/]

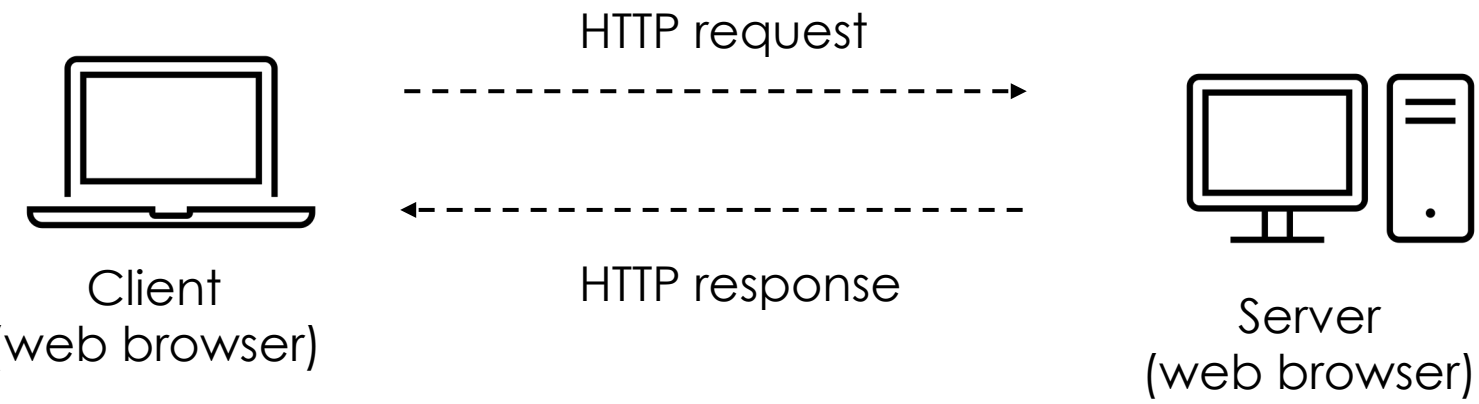
What is the Web?

*The **World Wide Web** -- commonly referred to as WWW, W3, or the Web -- is an **interconnected system of public webpages** accessible through the **Internet**.*

The Web is not the same as the Internet: the Web is one of many applications built on top of the Internet.

[https://developer.mozilla.org/en-US/docs/Glossary/World_Wide_Web]

World Wide Web (WWW)



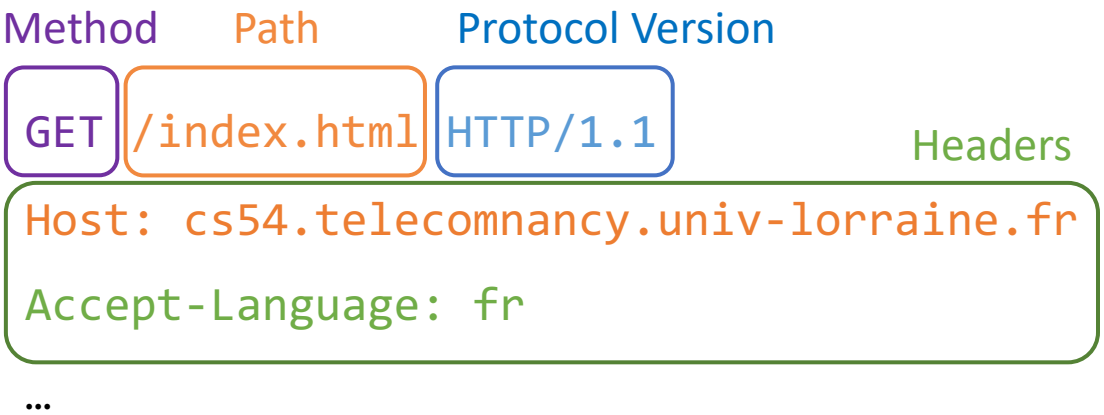
Uniform Resource Locator (URL):

`https://www.telecomnancy.univ-lorraine.fr/index.html`

Protocol — Domain and subdomains — Top Level Domain (TLD) — Path

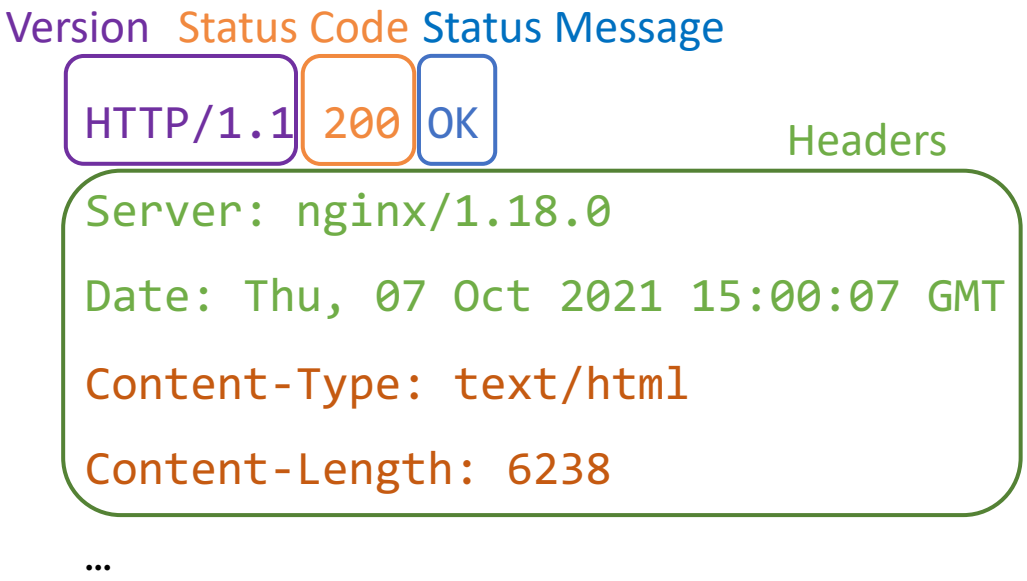
HyperText Transfer Protocol (HTTP)

- Protocol that defines:
  - Methods Request / HTTP Verbs : **GET**, **POST**, PUT, DELETE, HEAD, …



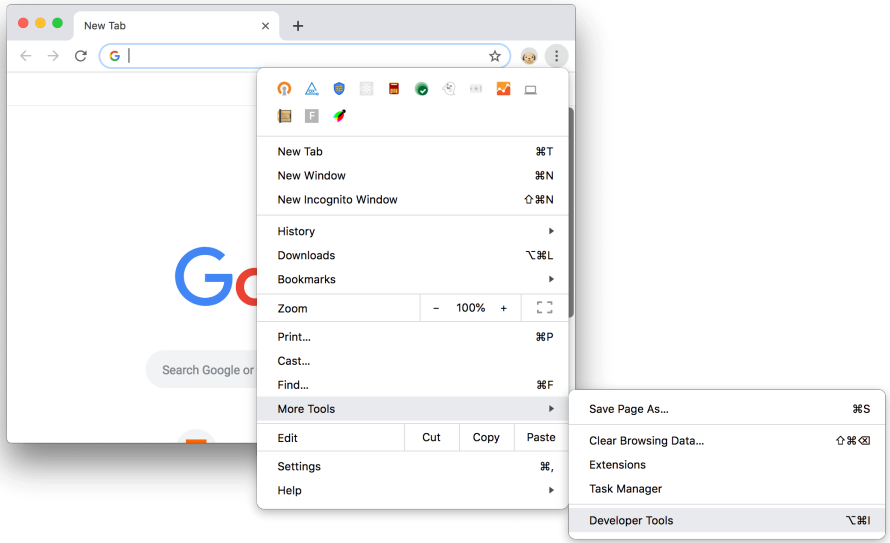
HyperText Transfer Protocol (HTTP) [cont.]

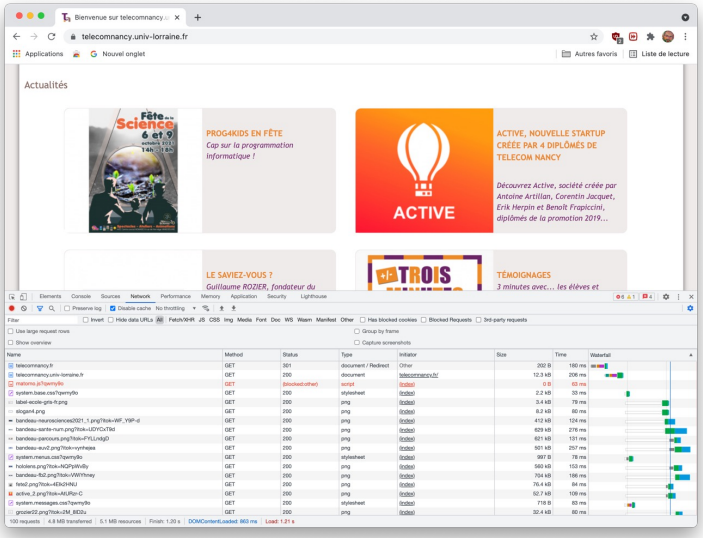
- Protocol that defines:
  - HTTP Responses



Web Developers Tools (demo)

- cURL : `$ curl -I http://www.google.fr/teapot`
- (Chrome) Web Developers Tools





HTTP Status Code

200 OK

301 Moved Permanently

302 Not Found

304 Not Modified

307 Temporary Redirect

401 Unauthorized

403 Forbidden

404 Not Found

418 I'm a Teapot

500 Internal Server Error

503 Service Unavailable

Information

Redirection

Client Errors

Server Errors

[https://datatracker.ietf.org/doc/html/rfc2616]
[https://developer.mozilla.org/en-US/docs/Web/HTTP/Status]

URL Parameters

```
GET /search?q=cats HTTP/1.1
Host: www.google.com
```

? : separator between path/parameters

`parameter_name = parameter_value`

```
GET /search?q=olivier+festor&tbm=isch HTTP/1.1
Host: www.google.com
```

& : separator between parameters

Names and values are *URL encoded!*

`Cha%C3%AEne%20pas%20tr%C3%A8s%20lisible%20%3B%29`

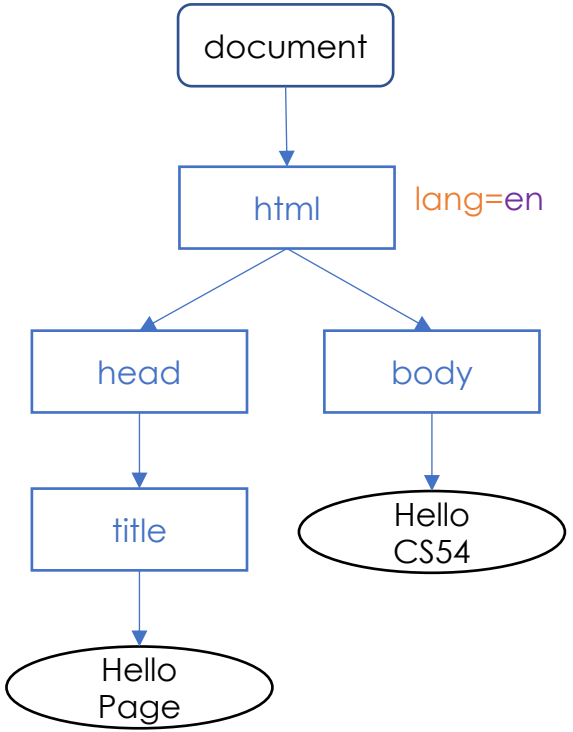
Hyper Text Markup Language (HTML)

- Markup language:
  - Tags: `<html> … </html>`
  - Attributes: `<html lang="fr"> … </html>`
- Anatomy of basic web page

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <title>Hello Page</title>
    </head>
    <body>
        Hello CS54
    </body>
</html>
```

Hyper Text Markup Language (HTML) [cont.]

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <title>Hello Page</title>
    </head>
    <body>
        Hello CS54
    </body>
</html>
```



Document Object Model (DOM)

Hyper Text Markup Language (HTML) [cont.]

- Live coding:
  - Local coding:
    - `file://` protocol
  - HTTP Server:
    - `python3 -m http.server 8080`
  - Visual Studio Code
    - IDE + **Live Server Extension**

Hyper Text Markup Language (HTML) [cont.]

- Paragraphs: `<p>…</p>`
- Headings: `<h1>,<h2>, …`
- Lists (unordered / ordered): `<ul>,<ol>,<li>, …`
- Table (rows, data): `<table>,<tr>, td>, …`
- Images `<img>`

```
<img alt="photo of the boss" src="./boss.jpg"/>
```

- **Links (relative / absolute + protocol)** `<a>`

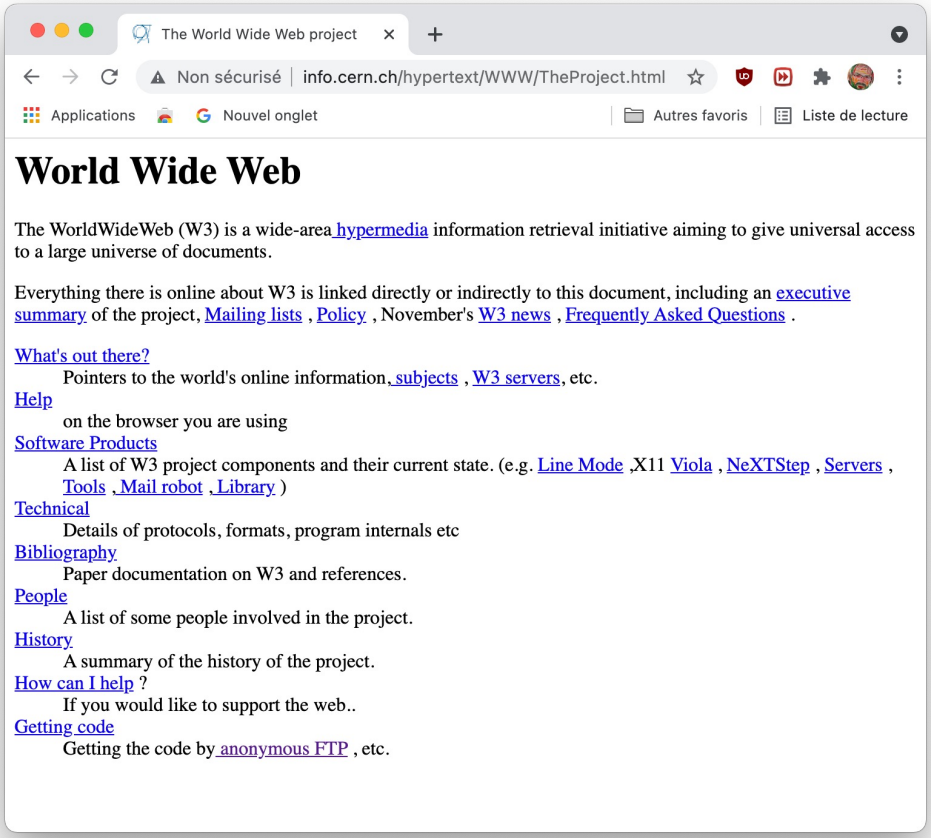
```
<a href="anotherworld.html"/>
```

```
<a href="https://thispersondoesnotexist.com/image"/>
```

  - Warning: Phishing issues

[http://info.cern.ch/hypertext/WWW/TheProject.html]

The First Web Site (1989)

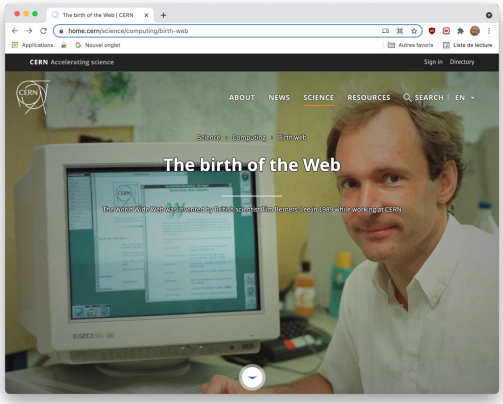


World Wide Web

The WorldWideWeb (W3) is a wide-area hypermedia information retrieval initiative aiming to give universal access to a large universe of documents.

Everything there is online about W3 is linked directly or indirectly to this document, including an executive summary of the project, Mailing lists , Policy , November's W3 news , Frequently Asked Questions .

What's out there?
Pointers to the world's online information, subjects , W3 servers, etc.
Help
on the browser you are using
Software Products
A list of W3 project components and their current state. (e.g. Line Mode ,X11 Viola , NeXTStep , Servers , Tools , Mail robot , Library)
Technical
Details of protocols, formats, program internals etc
Bibliography
Paper documentation on W3 and references.
People
A list of some people involved in the project.
History
A summary of the history of the project.
How can I help ?
If you would like to support the web..
Getting code
Getting the code by anonymous FTP , etc.



More information at
https://home.cern/science/computing/birth-web

Cascading Style Sheet (CSS) 101

```
a {
    color: blue;
    font-size: 5em;
}
a:hover {
    color: red;
    font-size: 5em;
}
```

Cascading Style Sheet (CSS) [cont.]

- Inlined as attributes of any tag using style
- Included in HTML file using <style> tag
- Included from an external file using <link>
  - `<link href="style.css" rel="stylesheet"/>`
- **Cascading styles**
- Properties `font-size: 24px;` there exists a lot of CSS properties
- Type selectors `p { … }` applied to all <p> attributes
- Class selectors `.warning { … }` applied to tag with attribute class="warning"
- ID selectors `#player { … }` applied to tag whose id= "player "

[https://developer.mozilla.org/fr/docs/Web/CSS/]

HTML Forms

Method, either post or get (more on that later)

HTML Form

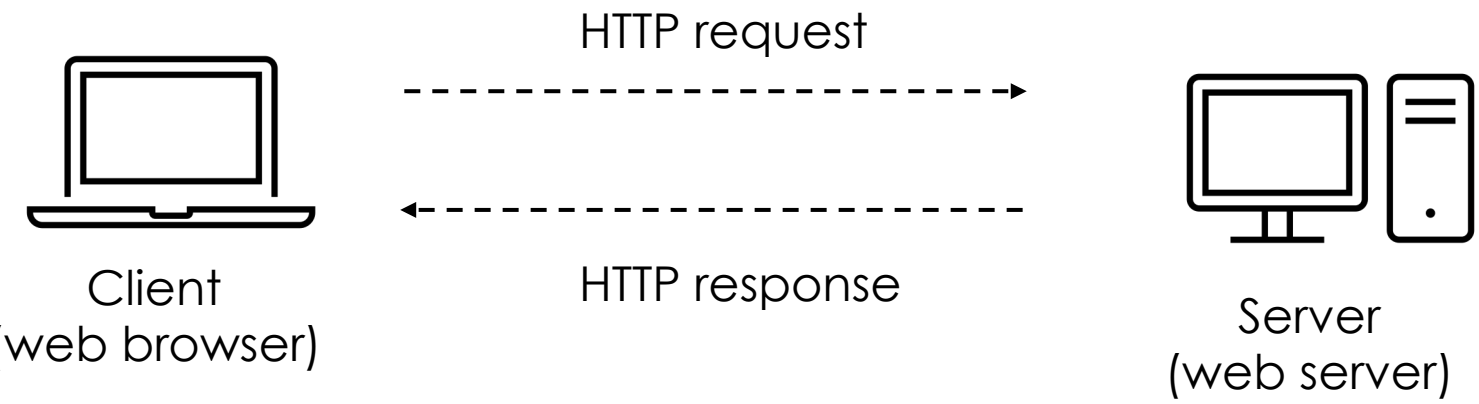
```
<form action="https://music.youtube.com/search" method="get">
    <input name="q" type="search"/>
    <input type="submit" value="Search"/>
</form>
```

Form Element

Coming next…

- Backend using Flask

Client / Server Model



HTML / CSS / (JavaScript)

Python / Flask

Flask

- Micro framework: Flask https://flask.palletsprojects.com/en/3.0.x/
  - **Lightweight Micro-framework**: Minimal core (request handling, routing) with flexible expansion options
  - **Built-in Development Tools**: Integrated server and debugger for easy testing.
  - **Jinja2 Templating**: Dynamic HTML rendering with a powerful template engine.
  - **Modular Design**: Choose libraries and tools as needed, with no rigid structure.
  - **Extensible**: Add features like databases or authentication via extensions.

```
$ python3 -m venv venv
$ source venv/bin/activate
$ pip install Flask
```

Flask– First web application

In a file named `app.py`

```
from flask import Flask

app = Flask(__name__)

@app.route('/')
def index():
    return 'Hello Web/BD'
```

In the terminal:

```
$ flask run
 * Environment: production
   WARNING: This is a development
   server. Do not use it in a
   production deployment.
   Use a production WSGI server
   instead.
 * Debug mode: off
 * Running on http://127.0.0.1:5000/
   (Press CTRL+C to quit)
```

Flask – How to run/configure the server

```
$ flask run
$ python -m flask run                      // alternatives
$ flask run -host=0.0.0.0 -port=8080   // passing options


// configuration using environment variables
$ export FLASK_APP=app.py
$ export FLASK_DEBUG=True          // debug + autorefresh
$ export FLASK_RUN_HOST=localhost
$ export FLASK_RUN_PORT=8080


$ pip install python-dotenv // then put all env variables
                                // in a file named .env
// or in the code of your main file
if __name__ == '__main__':
    app.run(host='0.0.0.0', port=8080, debug=True)
```

Routing

```
@app.route("/halloween")

@app.route("/halloween/")

@app.route("/monsters/<int:monster_id>")

@app.route("/some_full_path/<converter:variable_name>")
```

`string (default):` accepts any text without a slash

`int:` accepts positive integers

`float:` accepts positive floating point values

`path:` like string but also accepts slashes

`uuid:` accepts UUID strings

Routing - Example

```
@app.route("/monsters/<string:category>/<int:monster_id>")
def monster_by_id(category: str, monster_id: int):
    if category == "curcubitaceous" and monster_id == 666:
        return { "name": "Jack O'Lantern",
                 "id":666, "category": "curcubitaceous" }
    else:
        return {}
```

```
>>> GET /monsters/curcubitaceous/666 HTTP/1.1
```

Routing - Example

```
@app.route("/monsters/<string:category>/<int:monster_id>")
def monster_by_id(category: str, monster_id: int):
    if category == "curcubitaceous" and monster_id == 666:
        return { "name": "Jack O'Lantern",
                 "id":666, "category": "curcubitaceous" }
    else:
        return {}
```

automatically serialised as JSON Object

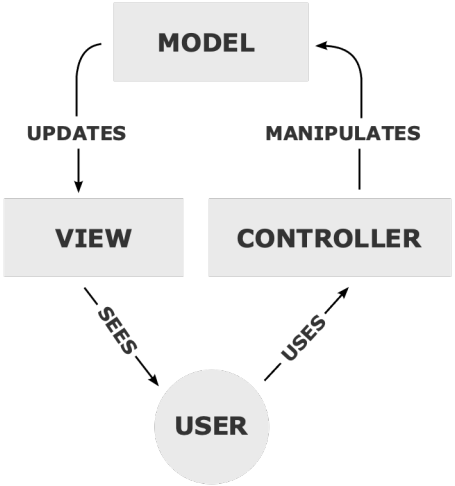
```
>>> GET /monsters/curcubitaceous/666 HTTP/1.1
```

JSON (JavaScript Object Notation)

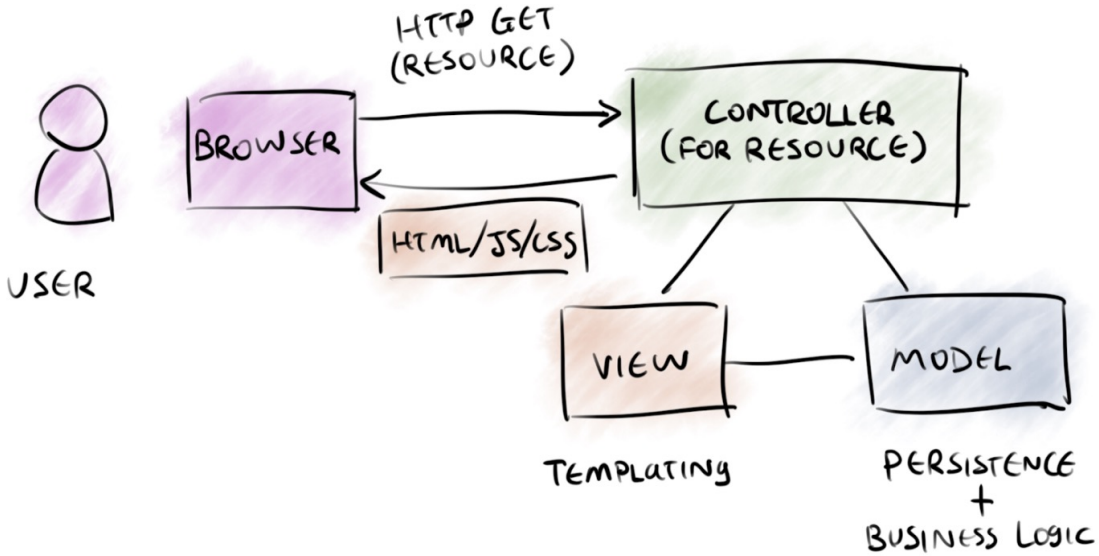
- Lightweight text-based data-interchange format
- Completely (programming) language independent
- Easy to understand, manipulate and generate
- Data types:
  - Strings (“hello”), numbers (integers, real), booleans (true of false), null
  - Arrays: ordered values wrapped in [and]
  - (JSON) object: Unordered key-value pairs wrapped in { }

```
{ "name": "Alice",
  "age": 30,
  "email": "alice@example.com",
  "is_active": true,
  "skills": ["Python", "Flask"],
  "address": {
    "street": "123 Main St",
    "city": "Wonderland",
    "zip": "12345"
  }
}
```

Model-View-Controller Pattern (MVC)



[Source: Wikipedia]



[Source: Medium, Robert Zhu]

What Not To Do

```
@app.route("/halloween")
def halloween():
    return """
<html>
<head>
<title>Halloween</title>
<style>
body { background: #000000; text-align:center;}
</style>
</head>
<body>
<img alt="jackolantern" width="100%" src="./static/halloween.jpg"/>
</body>
</html>
"""
```

Templates (powered by Jinja2)

In a file `templates/hello.html`

```
<!doctype html>
<html>
<title>Hello from Flask</title>
<body>
{% if name %}
   <h1>Hello {{ name }}!</h1>
{% else %}
   <h1>Hello, World!</h1>
{% endif %}
</body>
</html>
```

```
from flask import render_template

@app.route('/hello/')
@app.route('/hello/<login>')
def hello(login=None):
    return render_template('hello.html', name=login)
```

HTTP Methods

```
from flask import request

@app.route('/whois', methods=['GET', 'POST'])
def whois():
    if request.method == 'GET':
        return request.args.get('name', 'John Doe')
    else:
        return '' # do something else
```

```
>>> GET /whois?name=TheBoss HTTP/1.1
```

Forms (basic way of managing)

```
@app.route("/search", methods=["GET", "POST"])
def search():
    if request.method == "GET":
        return """<form action="/search" method="post">
                    <input name="q" type="search"/>
                    <input type="submit" value="Search"/>
                  </form>"""
    elif request.method == "POST":
        return do_the_job(request.form['q'])
    else:
      return {}
```

Static Resources / Redirect

- (by default) `'/static'` route is used for **static resources** (images, files, etc.) and associated with **`static/`** directory.

- The url of a resource can be obtained using the following code:

```
url_for('static', filename='style.css')
```

Redirection and Errors

```
from flask import abort, redirect, url_for

@app.route('/')
def index():
    return redirect(url_for('login'))

@app.route('/login')
def login():
    abort(401)
    this_is_never_executed()


from flask import render_template
@app.errorhandler(404)
def page_not_found(error):
    return render_template('page_not_found.html'), 404
```

Coming next…

- Persistence
- Sessions

SQLite3

- SQLite3 (https://www.sqlite.org/index.html):
  - « Public domain » Embedded database / SQL92 compliant (mostly)
- In the virtual machine provided by TN:

```
$ sudo apt-get install sqlite3
```

- Main commands:

  `.help` (your friends ;b)

  `.open` (open a database file – sqlite file format)

  `.quit` (exit SQLite)

  `.tables` (show all tables)

  `.schema` (show schema of all tables)

SQLite3

```
$ sqlite3 halloween.db
SQLite version 3.39.5 2022-10-14 20:58:05
Enter ".help" for usage hints.

sqlite> CREATE TABLE monsters(id integer, name varchar, height integer);
sqlite> .schema
CREATE TABLE monsters(id integer, name varchar, height integer);

sqlite> INSERT INTO monsters VALUES (1, 'Cthulhu', 47);
sqlite> INSERT INTO monsters VALUES (2, 'Nyarlathotep', 100);
sqlite> SELECT * FROM monsters ORDER BY height ASC;
1|Cthulhu|47
2|Nyarlathotep|100

sqlite> SELECT * FROM monsters ORDER BY height DESC;
2|Nyarlathotep|100
1|Cthulhu|47
```

Python / SQLite3

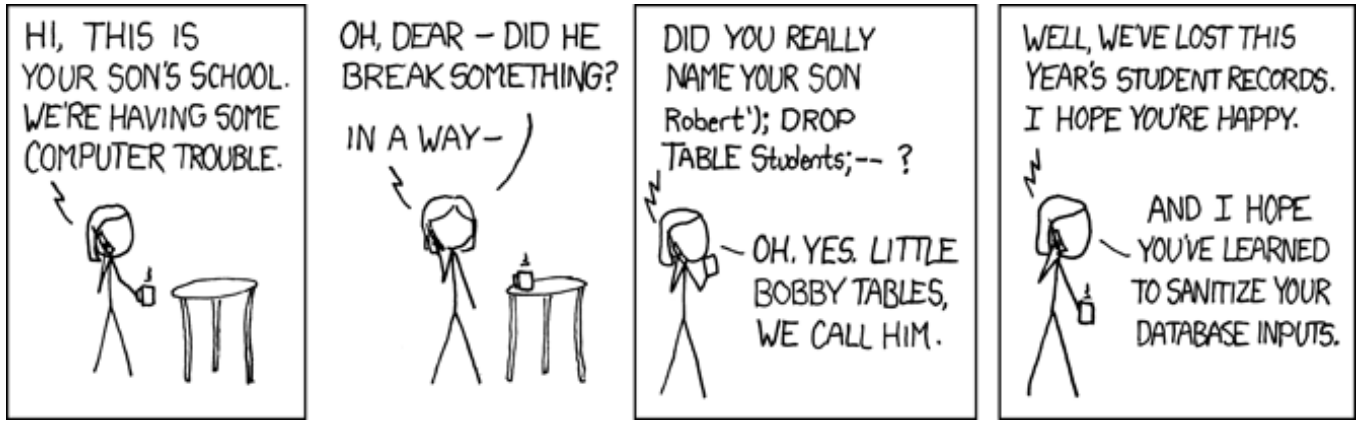
- https://docs.python.org/3/library/sqlite3.html

```
import sqlite3

con = sqlite3.connect('halloween.db')

cur = con.cursor()
# print(cur.execute('SELECT * FROM monsters')
# print(cur.fetchone())
# print(cur.fetchall())
for row in cur.execute('SELECT * FROM monsters ORDER BY height DESC’):
#   print(row)
    print(f'id: {row[0]}, name: {row[1]}, height: {row[2]}')

con.commit()
con.close()
```

Python / SQLite3 (cont.)

- What not to do (security issues: SQL injection):

```
monster_name = 'Cthulhu'
cur.execute("SELECT * FROM monsters WHERE name = '%s'" % monster_name)
# or:
cur.execute(f"SELECT * FROM monsters WHERE name = '{monster_name}'")
```



[Source: https://xkcd.com/327/]

Python / SQLite3 (cont.)

- How to do it:

```
# either:
cur.execute("SELECT * FROM monsters WHERE (?)", ('Cthulhu',))
# or:
cur.execute("SELECT * FROM monsters WHERE name=:mname", {"mname": 'Cthulhu'})
```

Beware! `('Cthulhu')` is `'Cthulhu'` so it is a string!
but you need to pass a python tuple!
`('Cthulhu',)` is a tuple in Python

A Simple App using Flask / SQLite3

```
from flask import Flask
from flask import g
import sqlite3

app = Flask(__name__)

DATABASE = 'halloween.db'

def get_db():
     db = getattr(g, '_database', None)
     if db is None:
          db = g._database = sqlite3.connect(DATABASE)
     return db

@app.teardown_appcontext
def close_connection(exception):
     db = getattr(g, '_database', None)
     if db is not None:
          db.close()
```

A Simple App using Flask / SQLite3 (cont.)

…

```
@app.route('/')
def status():
     return 'The server is running!'

@app.route('/monsters')
def all_monsters():
     c = get_db().cursor()
     c.execute("select * from monsters")
     return c.fetchall()

@app.route('/monsters/<int:monster_id>')
def one_monster(monster_id: str):
     c = get_db().cursor()
     c.execute("select * from monsters where id = ?", (monster_id,))
     return list(c.fetchone())
```

A Simple App using Flask / PostgreSQL

```
$ pip install psycopg2-binary
```

```
from flask import Flask
from flask import g
import psycopg2

app = Flask(__name__)

def get_db():
     db = getattr(g, '_database', None)
     if db is None:
          db = g._database = psycopg2.connect(database='postgres',
               host='localhost',user='postgres',password='mysecret',
               port='5432')
     return db
```

…

PostgreSQL (using Docker ;§)

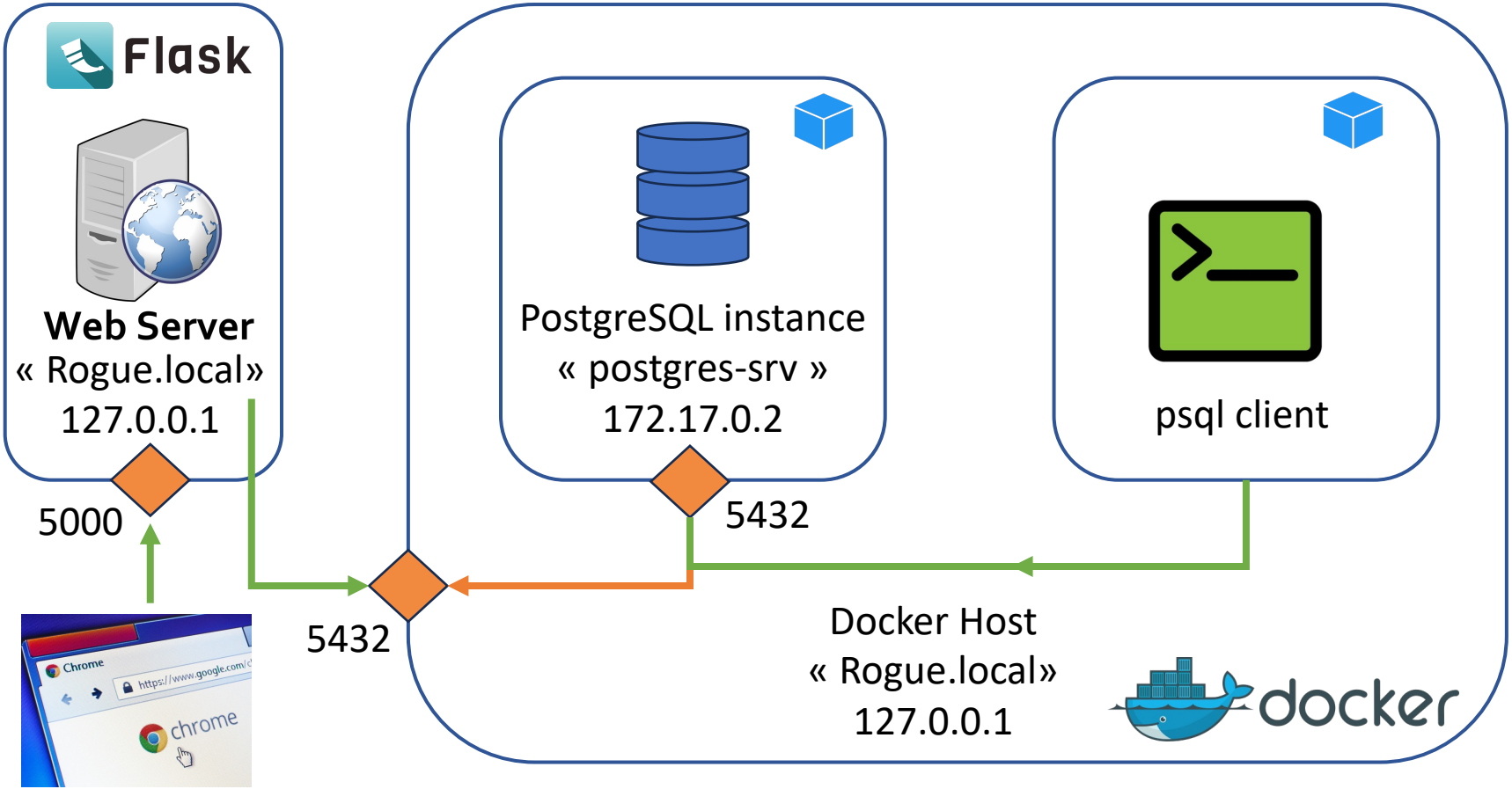
.SERVER in a container.

```
$ docker pull postgres:16
$ docker network create mynetwork
$ docker run –p 5432:5432 --name postgres-vm --network mynetwork
     --network-alias postgres-srv -e POSTGRES_PASSWORD=mysecret -d postgres
```

.CLIENT in a container.

```
$ docker ps
$ docker inspect postgres-vm | grep 'IPAddress'
 "IPAddress": "172.17.0.2",
$ docker run -it --network mynetwork --rm postgres psql -h postgres-srv -U
postgres
Password for user postgres:
psql (13.2 (Debian 13.2-1.pgdg100+1))
Type "help" for help.
postgres=# CREATE TABLE monsters(id integer, name varchar, height integer);
CREATE TABLE
postgres=# INSERT INTO monsters VALUES (1, 'Cthulhu', 47);
INSERT 0 1
```

Big Picture 😲


Flask
Web Server
« Rogue.local»
127.0.0.1
5000
PostgreSQL instance
« postgres-srv »
172.17.0.2
5432
psql client
5432
Docker Host
« Rogue.local»
127.0.0.1
docker
Chrome
https://www.google.com/
chrome

SQLAlchemy

https://www.sqlalchemy.org/

```
$ pip install SQLAlchemy
```

OPTIONAL

```
from sqlalchemy import create_engine, text

engine = create_engine('sqlite:///halloween.db')
# engine = create_engine('postgresql://user:password@host/database')

con = engine.connect()

rs = con.execute(text('SELECT * FROM book'))

for row in rs:
    print(row)

con.close()
```

SQLAlchemy / PostgreSQL

OPTIONAL

```
from sqlalchemy import create_engine, text

engine = create_engine('postgresql://postgres:mysecret@localhost/postgres')
con = engine.connect()

rs = con.execute(text('SELECT * FROM monsters'))
for row in rs:
   print(row)

con.close()
```

SQLAlchemy (cont.)

OPTIONAL

```
from sqlalchemy import create_engine, text

engine = create_engine('sqlite:///bookstore.db')
con = engine.connect()

rs = con.execute(text('DROP TABLE IF EXISTS book'))
rs = con.execute(text('CREATE TABLE book (id INTEGER PRIMARY_KEY,
                          title VARCHAR, primary_author VARCHAR)'))

statement = text('INSERT INTO book(id, title, primary_author) VALUES
                          (:id, :title, :primary_author)')
rs = con.execute(statement, {'id':1, 'title':'The Silmarillion',
                                     'primary_author':'Tolkien' })
for row in con.execute(text('SELECT * FROM book')):
   print(row)
con.close()
```

Flask-SQLAlchemy

OPTIONAL

https://flask-sqlalchemy.palletsprojects.com/en/3.1.x/

```
$ pip install flask_sqlalchemy
```

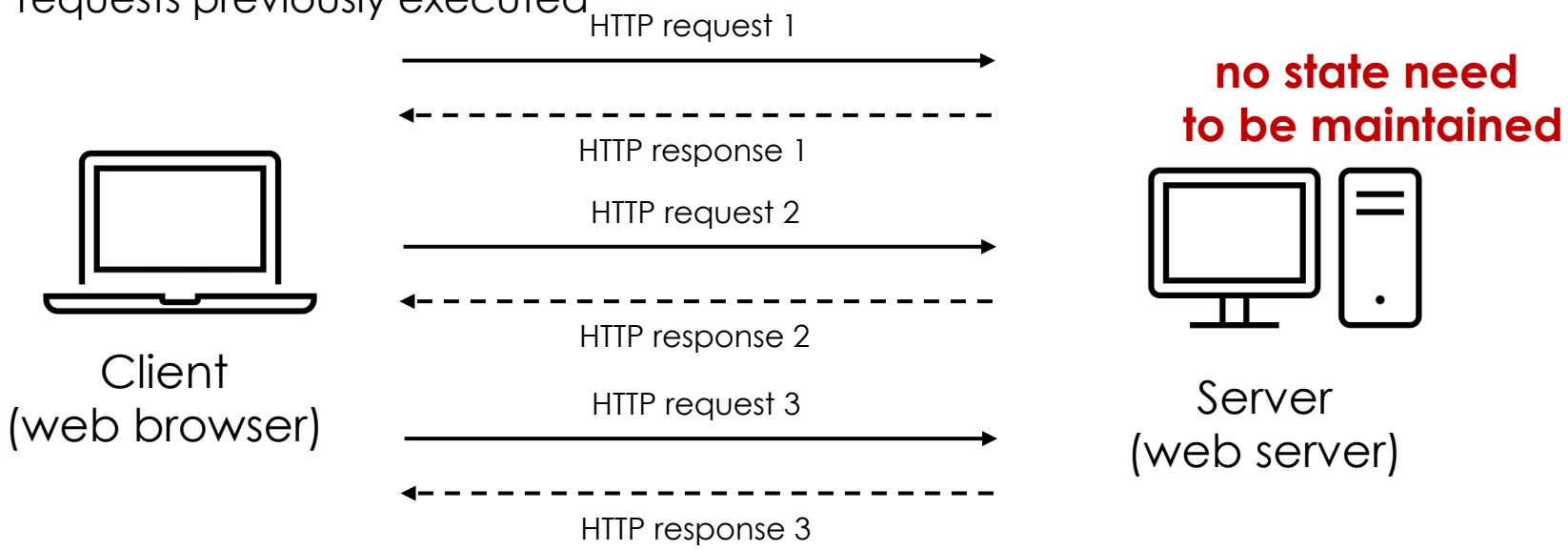
```
from flask import Flask
from sqlalchemy import text
from flask_sqlalchemy import SQLAlchemy

app = Flask(__name__)
app.config['SQLALCHEMY_DATABASE_URI'] = 'sqlite:////tmp/halloween.db'
db = SQLAlchemy(app)

@app.route('/')
def status():
    return 'The server is running!'

@app.route('/monsters')
def all_monsters():
    c = db.session
    res = c.execute(text("select * from monsters"))
    return str(res.fetchall())
```

To go further…

- Me want Cookie!

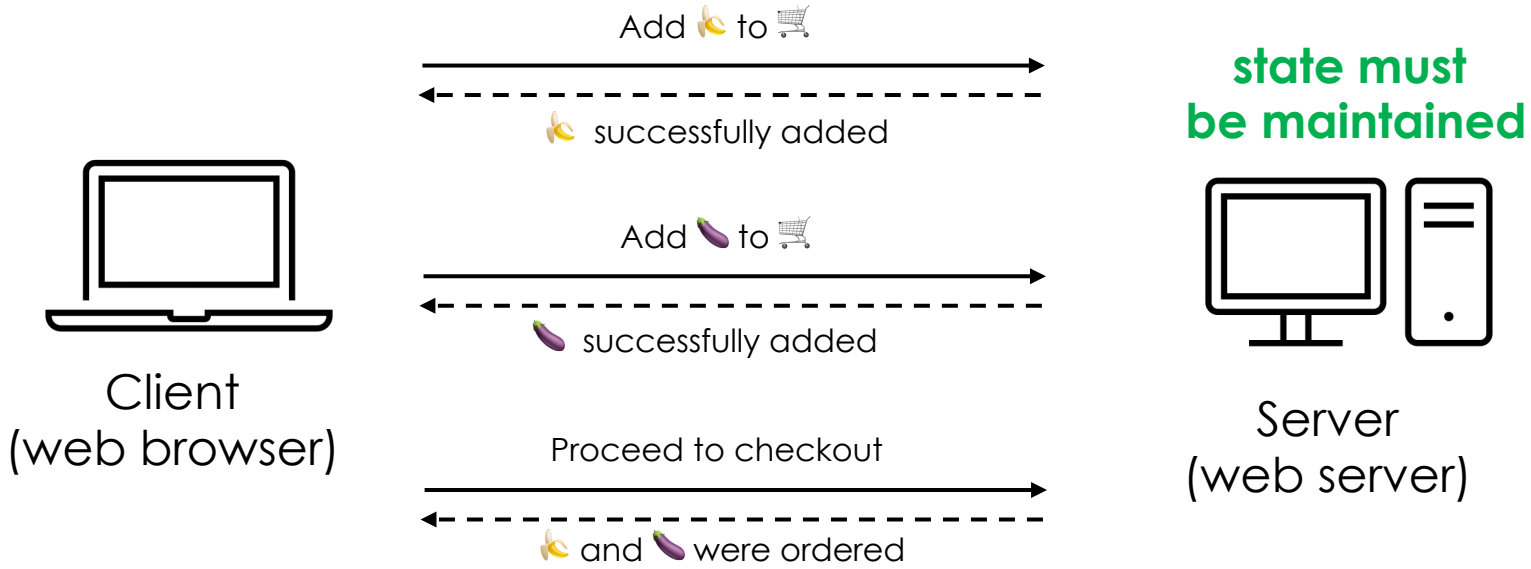
HTTP/1: A stateless protocol

- Protocol does not require server to retain information over successive requests from the same client
- Each request is executed independently without any knowledge of the requests previously executed



Web Application are often stateful

- An application must often track user's progress from page to page over successive requests
- i.e. shopping cart, user logged in or not, user preferences, etc

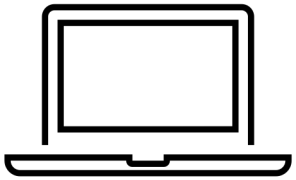


Session

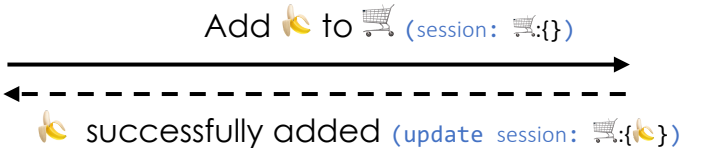
- A **session** is used to store information related to a user, across different requests, as they interact with a web app
- The data stored for a session are generally considered **temporary data**, ass the session will **eventually expire**
- Two kinds of session in web development:
  - Client-side: sessions are stored client-side:
    data are (re)sent with every requests
  - Server-side: sessions are stored server-side:
    an identifier is usually attached to every requests

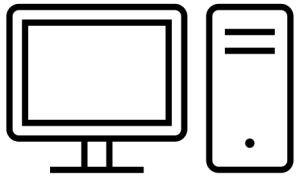
Client-side Session

- Client-side: sessions are stored client-side
- data are *usually* (re)sent with every requests



Client
(web browser)





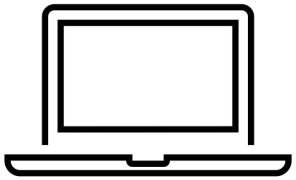
Server
(web server)

Data kept at client

session: 🛒:{🍌, 🍆}

Server-side Session

- Sessions are stored server-side
- an identifier is *usually* attached to every requests to retrieve the session data at server side



Client
(web browser)

Add 🍌 to 🛒 (session id: `mSjGCTfn8w`)

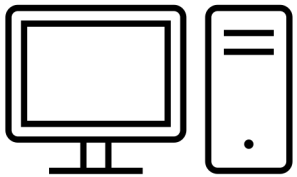
🍌 successfully added

Add 🍆 to 🛒 (session id: `mSjGCTfn8w`)

🍆 successfully added

Proceed to checkout (session id: `mSjGCTfn8w`)

🍌 and 🍆 were ordered



Server
(web server)

Data kept at client

session id: `mSjGCTfn8w`

Data kept at server

```
p0ZoB1FwH6 -> 🛒:{}
mSjGCTfn8w -> 🛒:{🍌, 🍆}
yt4Xx5nHMB -> 🛒:{🎁}
```

How to send some data with every requests?
Using parameter(s)

```
GET /add_article?session_id=mSjGCTfn8w&article=🍌 HTTP/1.1
Host: www.nozama.com

GET /add_article?session_id=mSjGCTfn8w&article=🍆 HTTP/1.1
Host: www.nozama.com
…
```

- Tedious but it works!

Using HTTP Cookies!

https://datatracker.ietf.org/doc/html/rfc6265

```
GET /browse_article HTTP/1.1
Host: www.nozama.com

HTTP/1.1 200
Content-type: text/html
Set-Cookie: session_id=mSjGCTfn8w
…

GET /add_article?article=🍌 HTTP/1.1
Host: www.nozama.com
Cookie: session_id=mSjGCTfn8w

HTTP/1.1 200
Content-type: text/html
…

GET /add_article?article=🍆 HTTP/1.1
Host: www.nozama.com
Cookie: session_id=mSjGCTfn8w

HTTP/1.1 200
Content-type: text/html
```

- Client request #1
- Server response #1
- Client request #2
- Server response #2
- Client request #3
- Server response #3

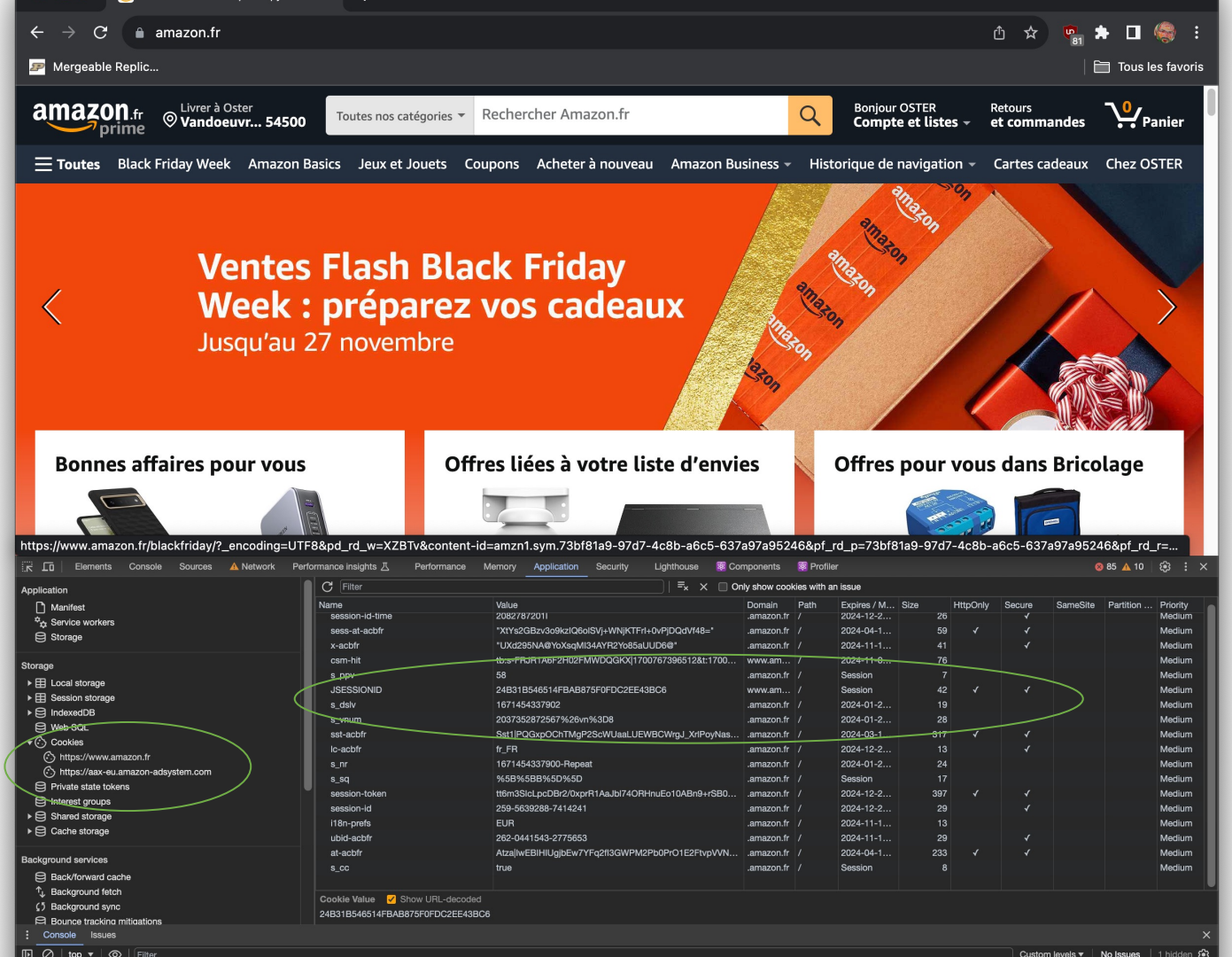
HTTP Cookies

https://developer.mozilla.org/fr/docs/Web/HTTP/Cookies

- Cookie:
  - Set of key/value pairs exchanged between HTTP client and HTTP server
  - Stored at client side (by the web browser)
  - Sent with every request (according to the domain/path attributes)
- Cookie Attributes:
  - Expiration date
  - Path
  - Domain
  - `Secure; HttpOnly;`

```
Set-Cookie: session_id=
mSjGCTfn8w;expires=Sun, 17-Jan-2038
23:08:01 GMT;path=/;domain=.nozama.com
```

- Expiration:
  - When expiration date has passed
  - When removed by client or update to old date by server
  - Browser is closed and cookie is not persistent


amazon.fr
Mergeable Replic...
Tous les favoris
amazon.fr prime
Livrer à Oster
Vandoeuvr... 54500
Toutes nos catégories
Rechercher Amazon.fr
Bonjour OSTER
Compte et listes
Retours
et commandes
0
Panier
Toutes
Black Friday Week
Amazon Basics
Jeux et Jouets
Coupons
Acheter à nouveau
Amazon Business
Historique de navigation
Cartes cadeaux
Chez OSTER
Ventes Flash Black Friday Week : préparez vos cadeaux
Jusqu'au 27 novembre
Bonnes affaires pour vous
Offres liées à votre liste d'envies
Offres pour vous dans Bricolage
https://www.amazon.fr/blackfriday/?_encoding=UTF8&pd_rd_w=XZBTv&content-id=amzn1.sym.73bf81a9-97d7-4c8b-a6c5-637a97a95246&pf_rd_p=73bf81a9-97d7-4c8b-a6c5-637a97a95246&pf_rd_r=...
Elements
Console
Sources
Network
Performance insights
Performance
Memory
Application
Security
Lighthouse
Components
Profiler
Application
Manifest
Service workers
Storage
Storage
Local storage
Session storage
IndexedDB
Web SQL
Cookies
https://www.amazon.fr
https://aax-eu.amazon-adsystem.com
Private state tokens
Interest groups
Shared storage
Cache storage
Background services
Back/forward cache
Background fetch
Background sync
Bounce tracking mitigations
Filter
Only show cookies with an issue
Name
Value
Domain
Path
Expires / M...
Size
HttpOnly
Secure
SameSite
Partition ...
Priority
session-id-time
sess-at-acbfr
x-acbfr
csm-hit
s_ppv
JSESSIONID
s_dslv
s_vnum
sst-acbfr
lc-acbfr
s_nr
s_sq
session-token
session-id
i18n-prefs
ubid-acbfr
at-acbfr
s_cc
Medium
Cookie Value
Show URL-decoded
24B31B546514FBAB875F0FDC2EE43BC6
Console
Issues

Cookies using Flask

https://flask.palletsprojects.com/en/3.0.x/quickstart/#cookies

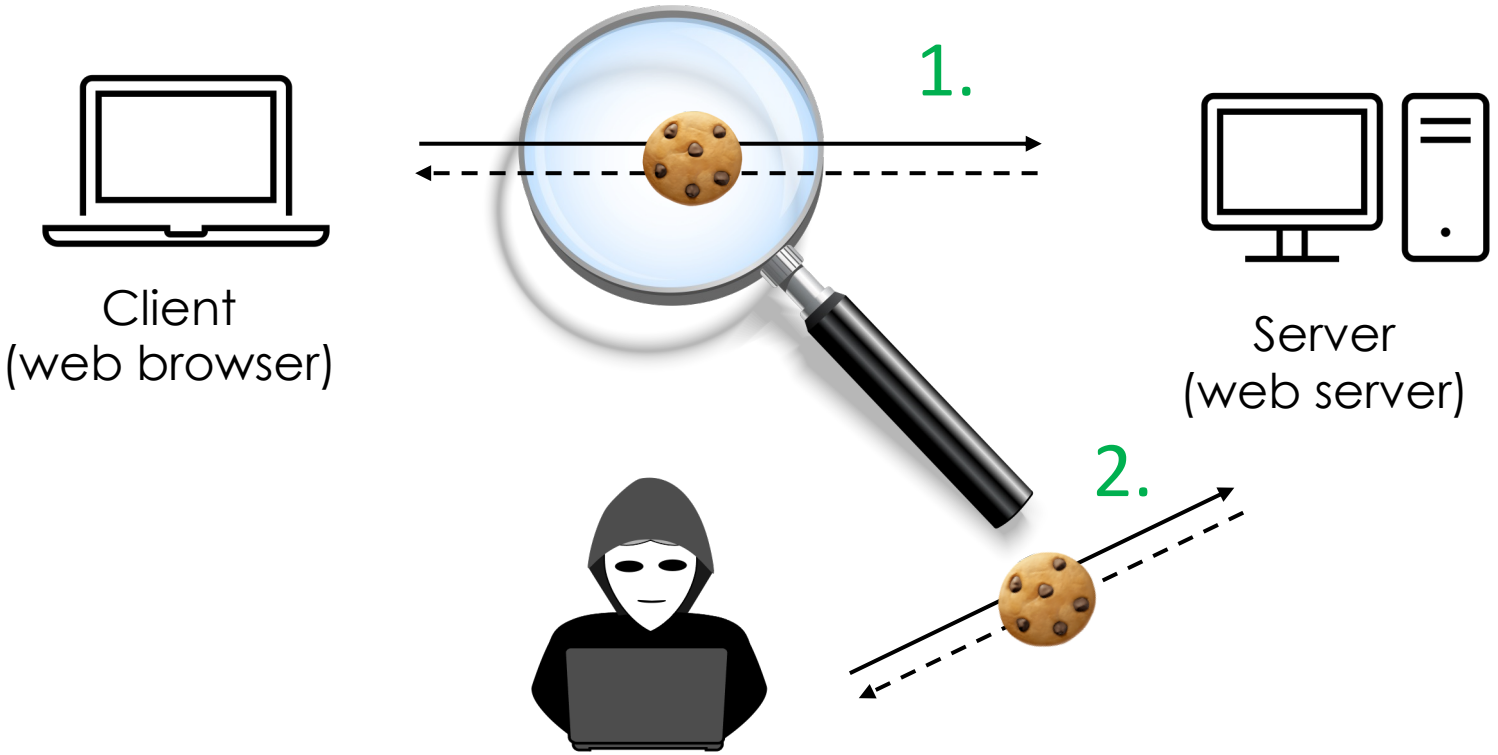
```
@app.route('/some_route')
def route_that_set_a_cookie():
  # do something

  some_data = 'somevalue'
  resp = make_response(render_template('mynice.html', …)

  resp.set_cookie('mycookie', some_data)
  return resp


@app.route('/some_other_route')
def route_that_read_a_cookie():
  # do something

  some_data = request.cookies.get('mycookie')
  return …
```

Server-side Session using Flask

https://flask.palletsprojects.com/en/3.0.x/quickstart/#sessions

```
from flask import Flask, session
app = Flask(__name__)
app.secret_key = 'SOME_SECRET'
app.config['PERMANENT_SESSION_LIFETIME'] = timedelta(minutes=1)

@app.route('/')
def route_that_store_a_value_in_a_session():
    session['email'] = 'johndoe@telecomnancy.eu'
    return 'Hello CS54'


@app.route('/another')
def route_that_read_a_value_from_a_session():
        return session['email']
```

Session Cookie Hijacking



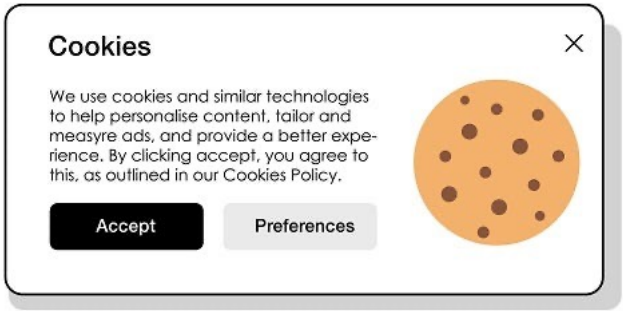
Flask-Session

https://flask-session.readthedocs.io/en/latest/

Server-side Session with Storage

```
from flask import Flask, session
from flask_session import Session

app = Flask(__name__)
SESSION_TYPE = 'filesystem'
app.config.from_object(__name__)
Session(app)

@app.route('/set/')
def set():
   session['somekey'] = 'somevalue'
   return 'ok'

@app.route('/get/')
def get():
    return session.get('somekey', 'value if not set')
```

Cookies Policy – Do you know why?



Quel est le cadre juridique applicable ?

L'article 5(3) de la directive 2002/58/CE modifiée en 2009 pose le principe :

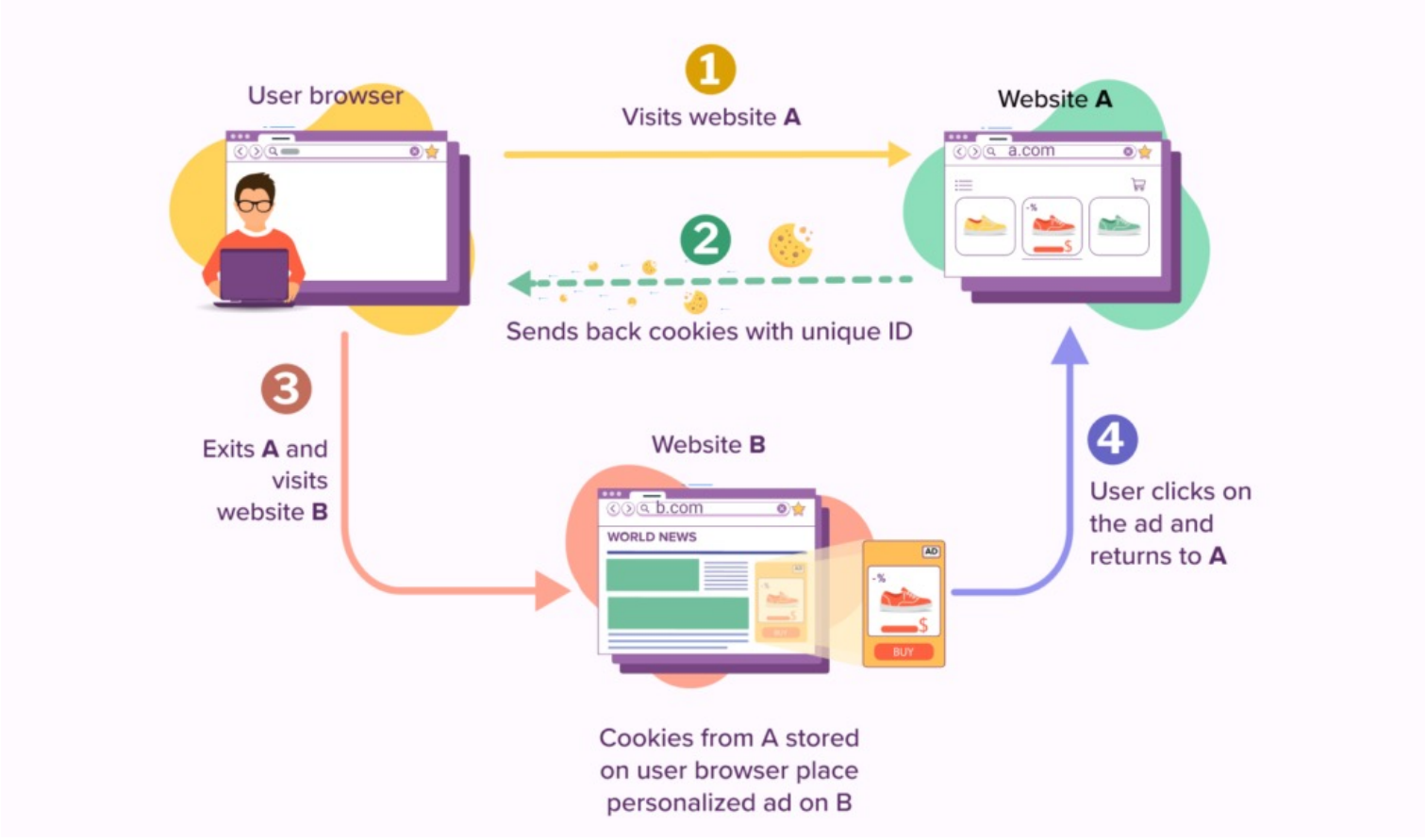
- d'un consentement préalable de l'utilisateur avant le stockage d'informations sur son terminal ou l'accès à des informations déjà stockées sur celui-ci ;
- sauf si ces actions sont strictement nécessaires à la fourniture d'un service de communication en ligne expressément demandé par l'utilisateur ou ont pour finalité exclusive de permettre ou faciliter une communication par voie électronique.

L'article 82 de la loi Informatique et Libertés transpose ces dispositions en droit français.

La CNIL rappelle que le consentement prévu par ces dispositions renvoie à la définition et aux conditions prévues aux articles 4(11) et 7 du RGPD. Il doit donc être libre, spécifique, éclairé, univoque et l'utilisateur doit être en mesure de le retirer, à tout moment, avec la même simplicité qu'il l'a accordé.

Afin de rappeler et d'expliciter le droit applicable au dépôt et à la lecture de traceurs dans le terminal de l'utilisateur, la CNIL a adopté le 17 septembre 2020 des lignes directrices, complétées par une recommandation visant notamment à proposer des exemples de modalités pratiques de recueil du consentement.

Third-Party Cookie Retargeting



To go further…

- Views and Blueprints
- SQLAlchemy (Python ORM)
- REST API (Marshmallow)
- Forms (WTF)
- Security (Authentication)
- Deployment (WSGI)
- …