

# ARCHITECTURE

Michael  
X  NATIS

# L'architecture, c'est quoi ?





**Compétence demandée :**  
**Comprendre l'architecture des échanges +  
architecture des applications**

1. Architecture de l'infrastructure
2. Architecture du transport
3. Architecture de l'application

Client

HTML

css

JS

Livraison  
de la  
page

données  
JSON

REST  
API

Back  
end

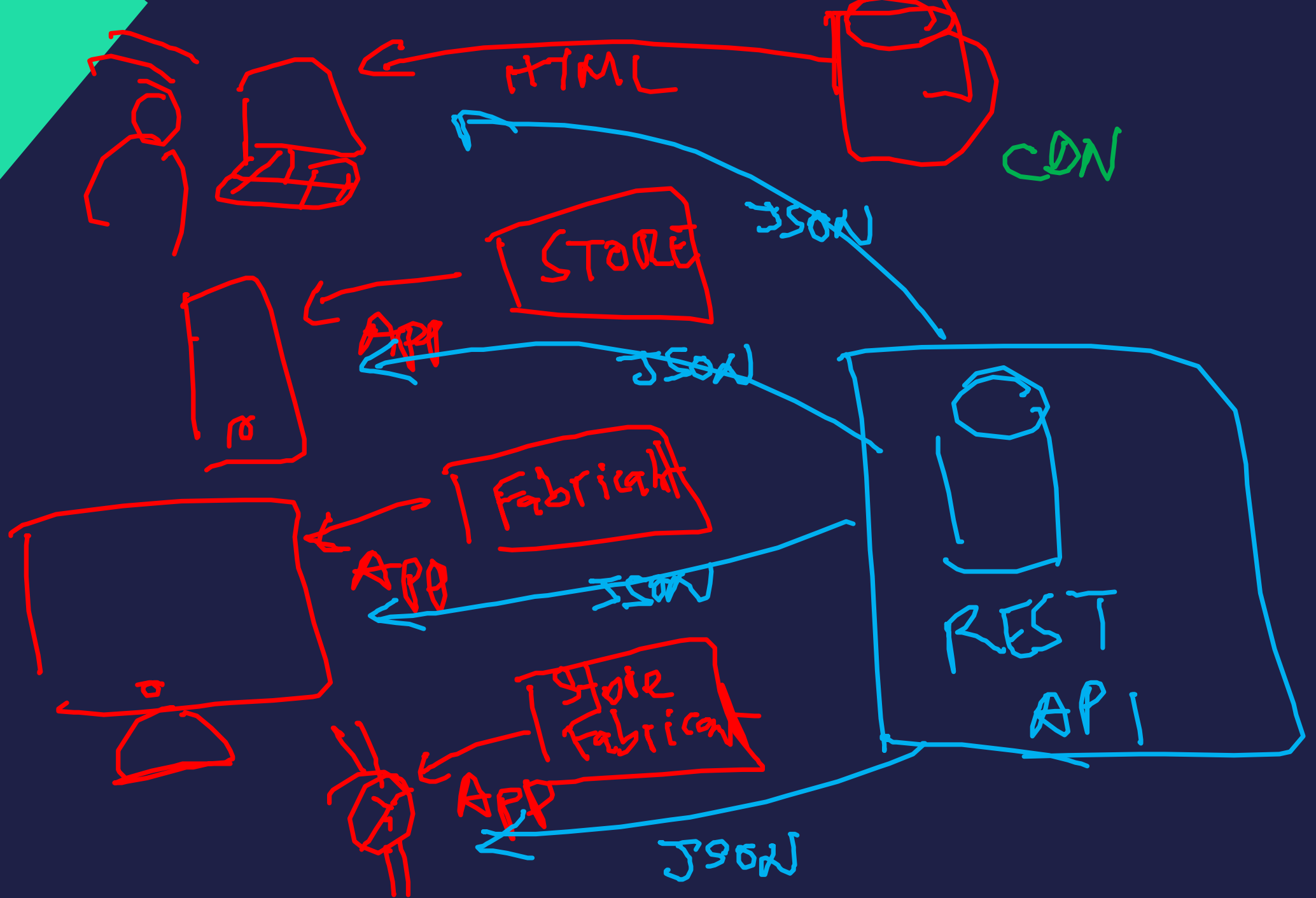
Client

html  
css  
...

PHP

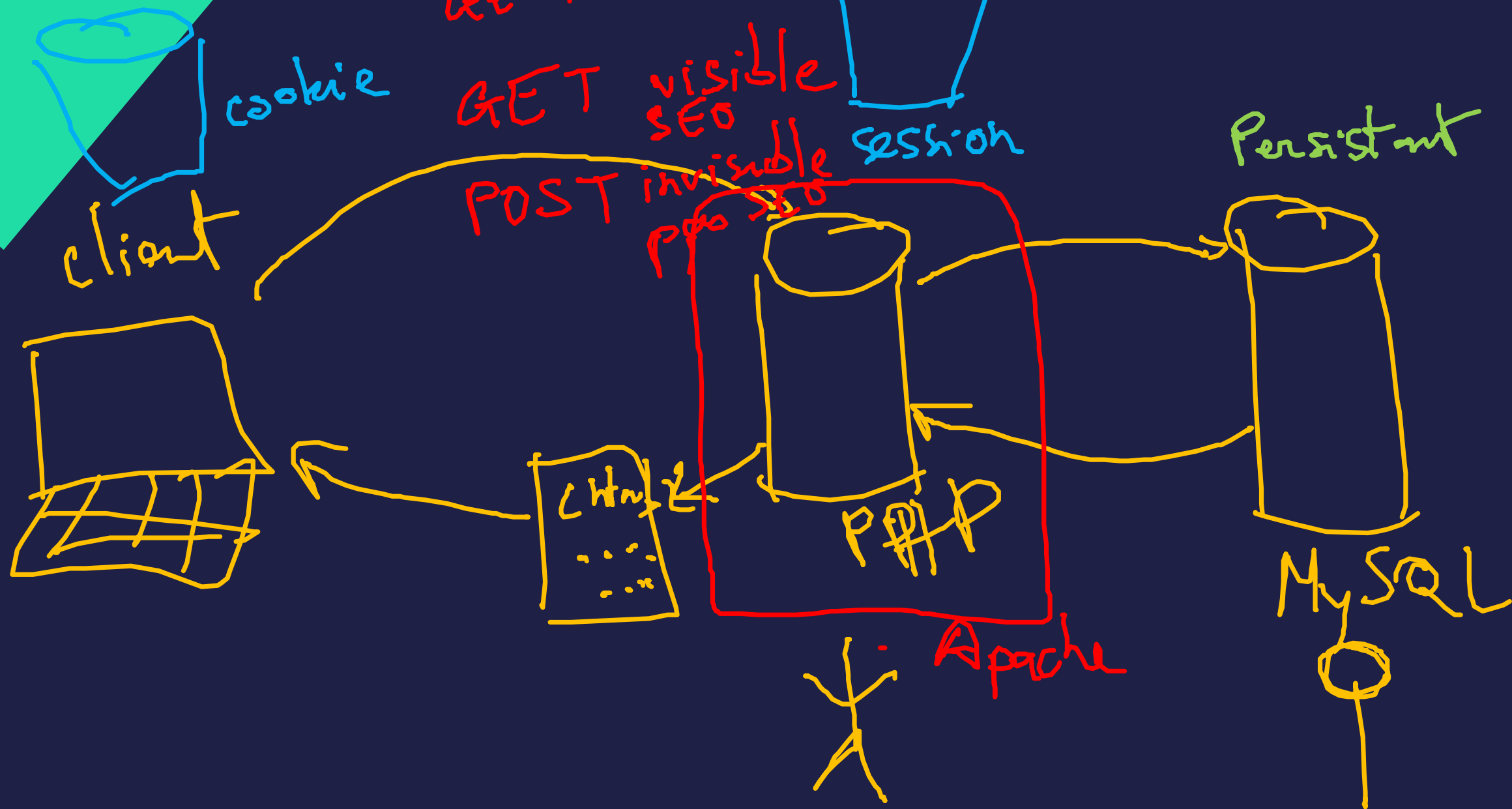
serveur

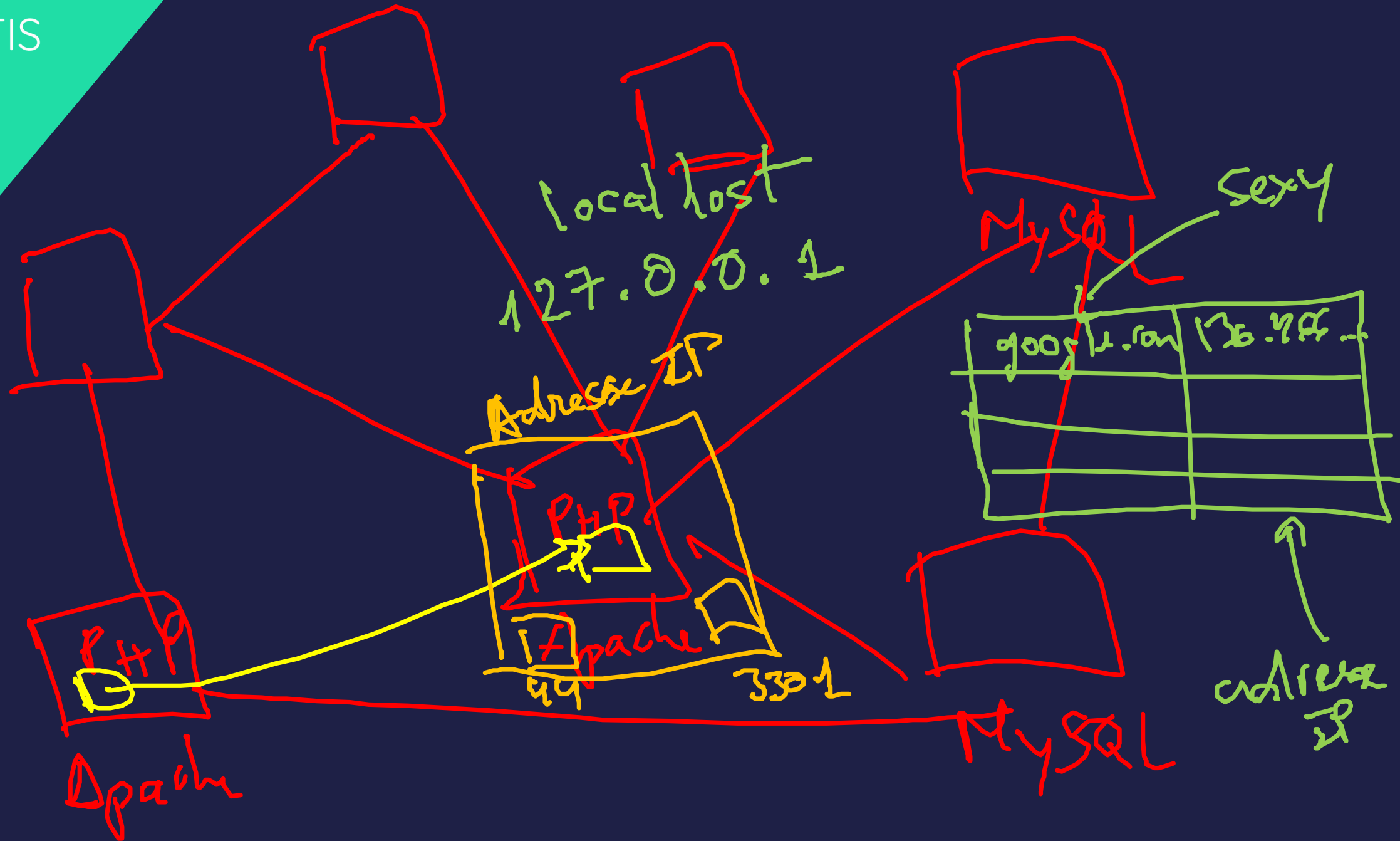
données











Une page statique : une page qui est renvoyée au client telle qu'elle a été stockée sur le serveur

Une page dynamique: une page qui  
est **générée à la volée par le**  
**serveur**

Une page statique : une page qui est **renvoyée au client telle qu'elle a été stockée sur le serveur**





<http://www.super-site.com>

X NATIS





4 super héros  
sont là pour  
vous !  
(très important)



Chrome



DNS



Apache



MySQL



Chrome





Chrome

DNS

Je cherche la page  
pour 'vélo'



Chrome

DNS

Je cherche la page  
pour 'vélo'

HTML + JavaScript + CSS



Chrome

Je cherche la page  
pour 'vélo'

HTML + JavaScript + CSS

super-site.com



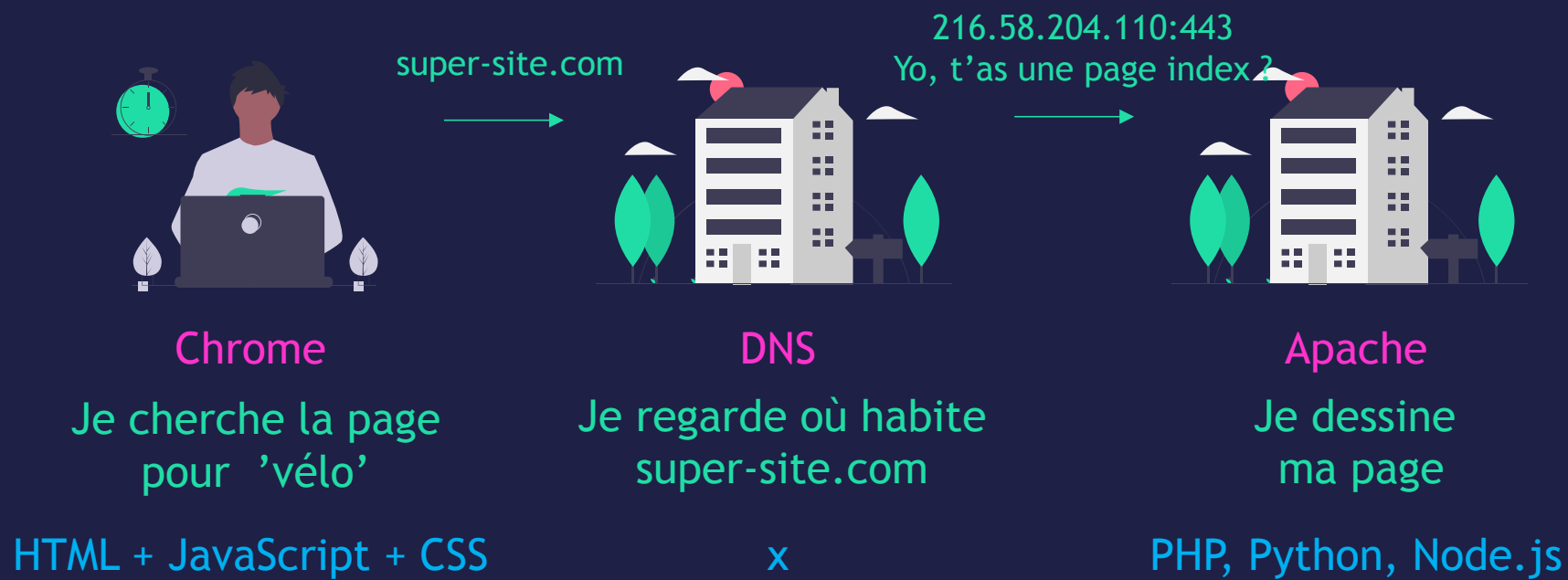
DNS

Je regarde où habite  
super-site.com

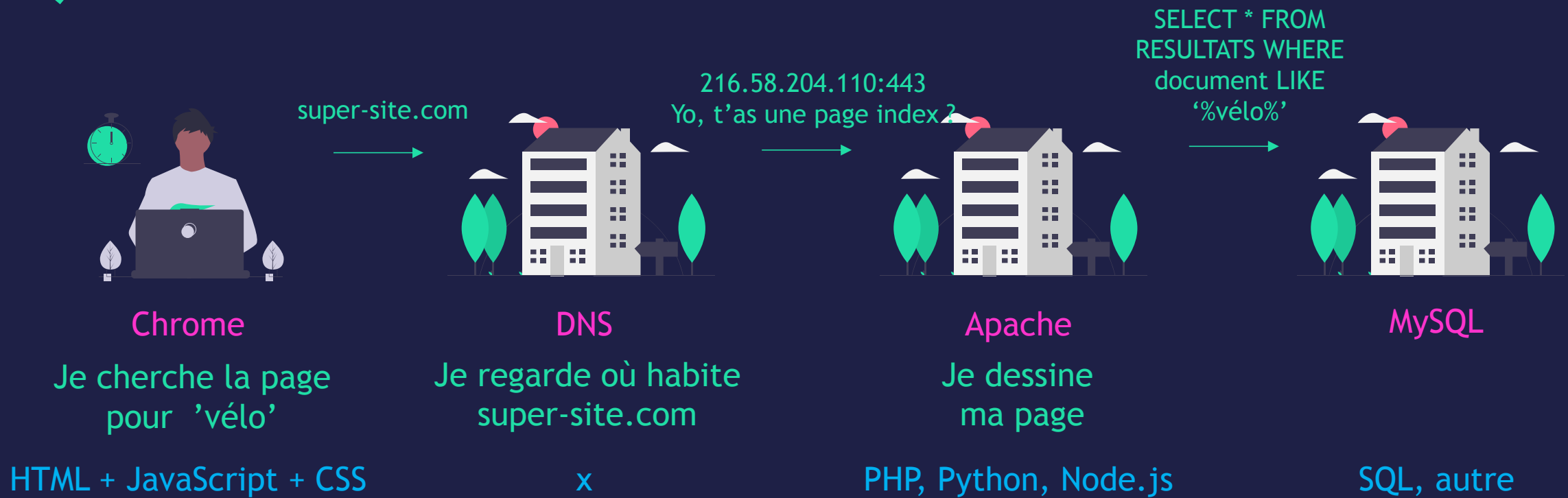
x

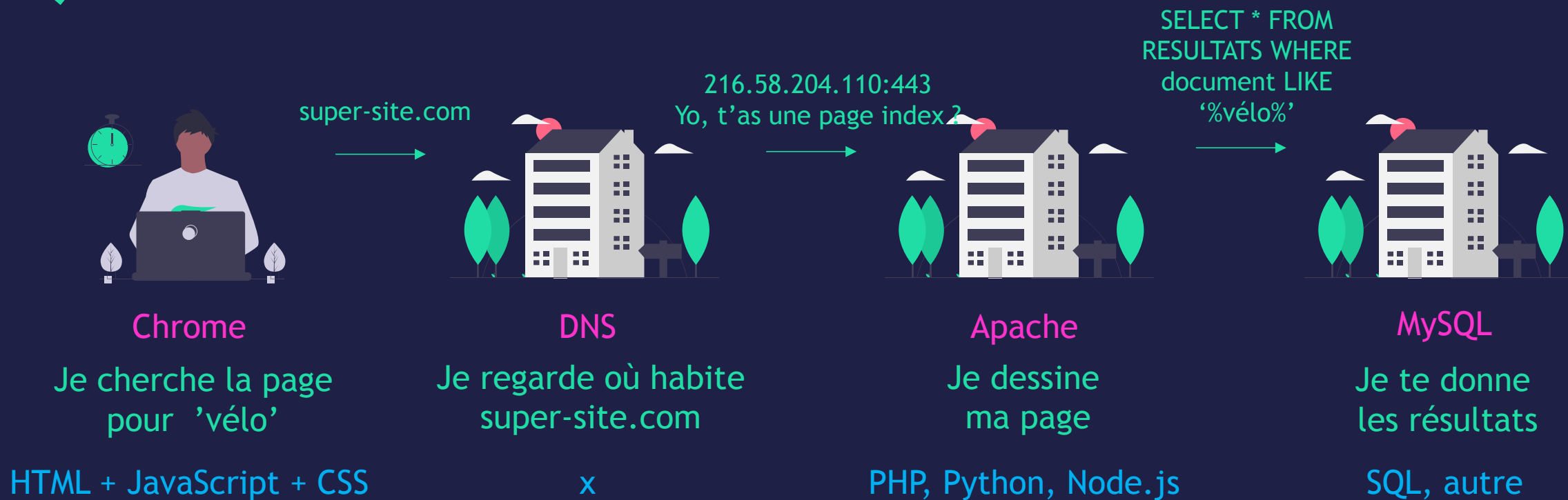


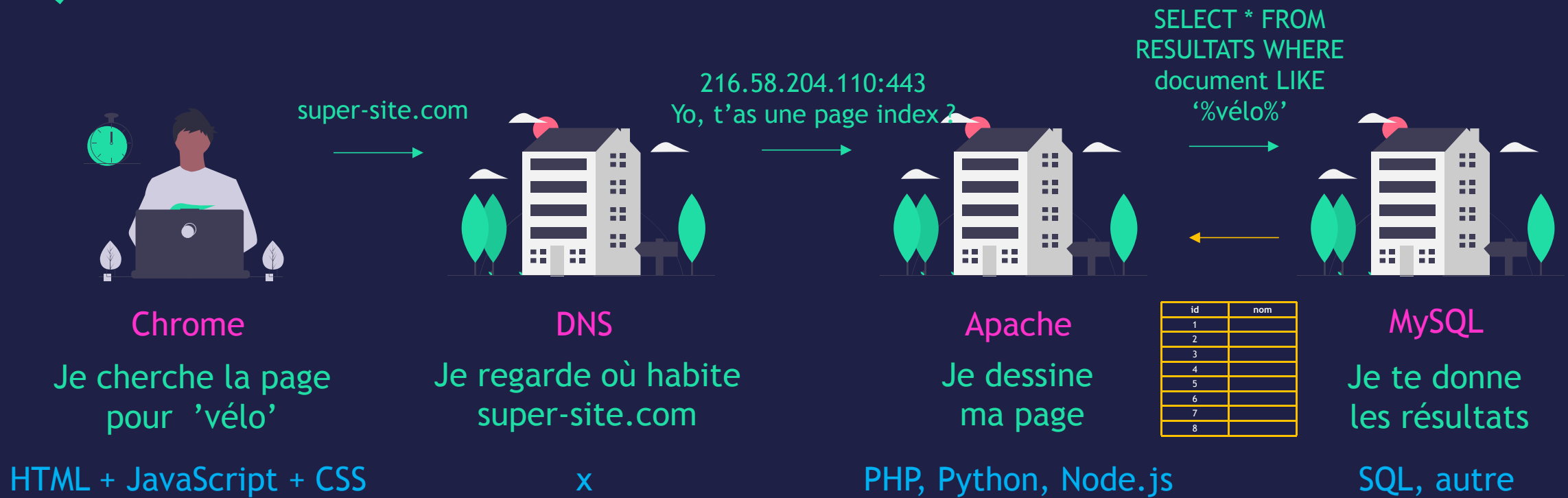


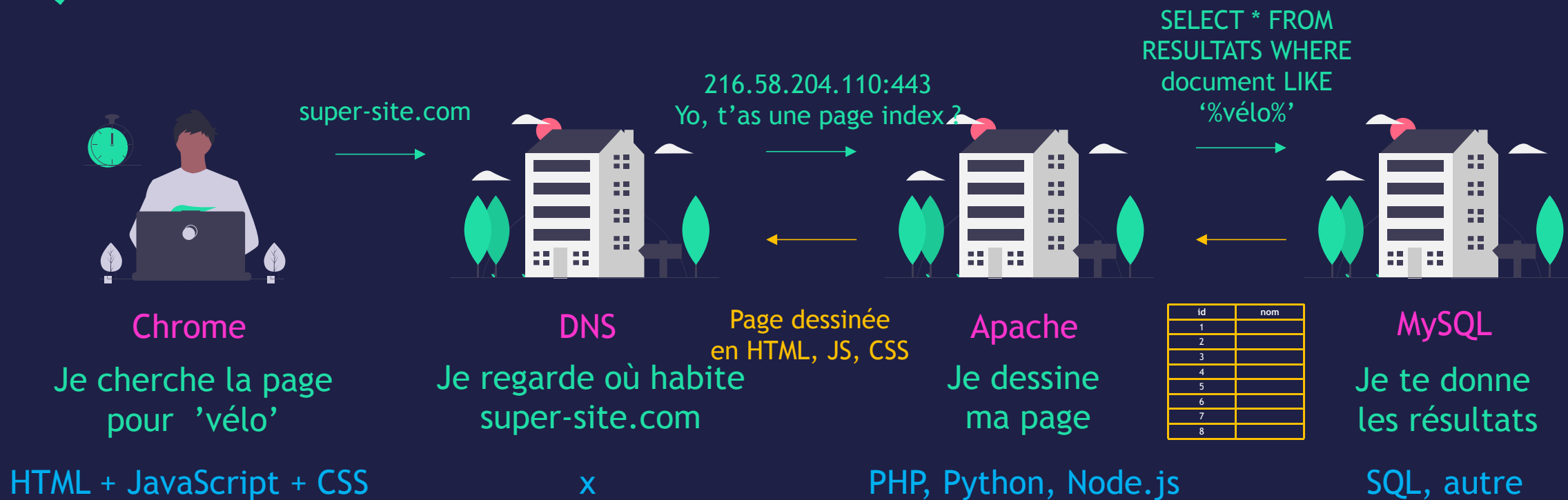


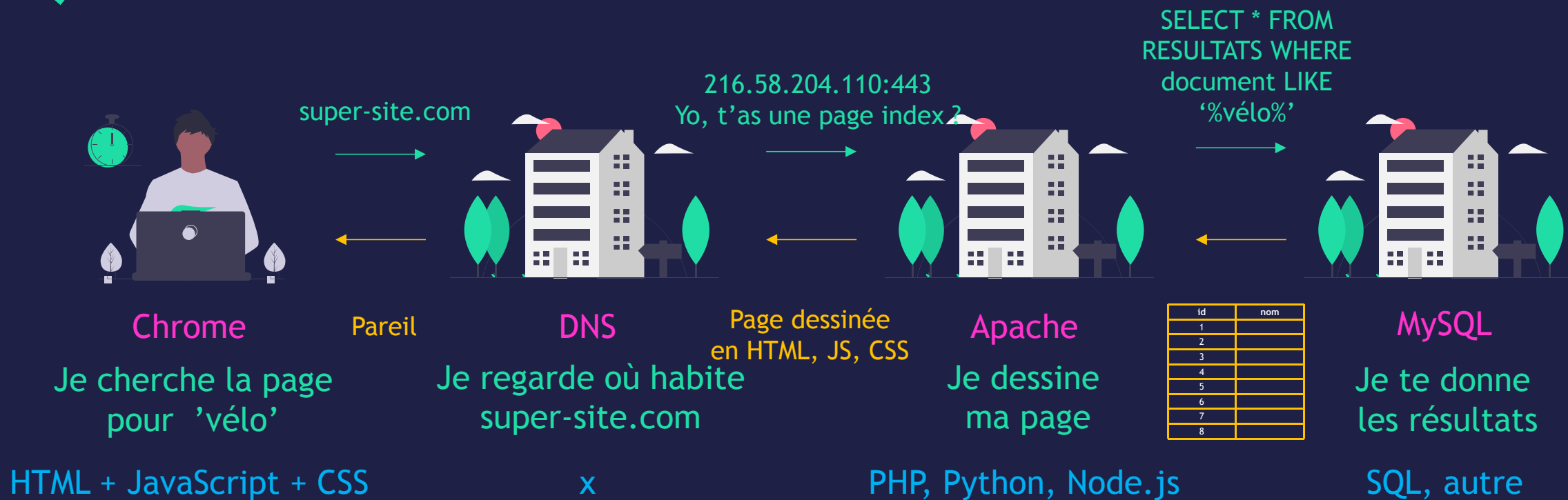






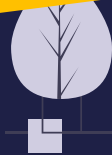




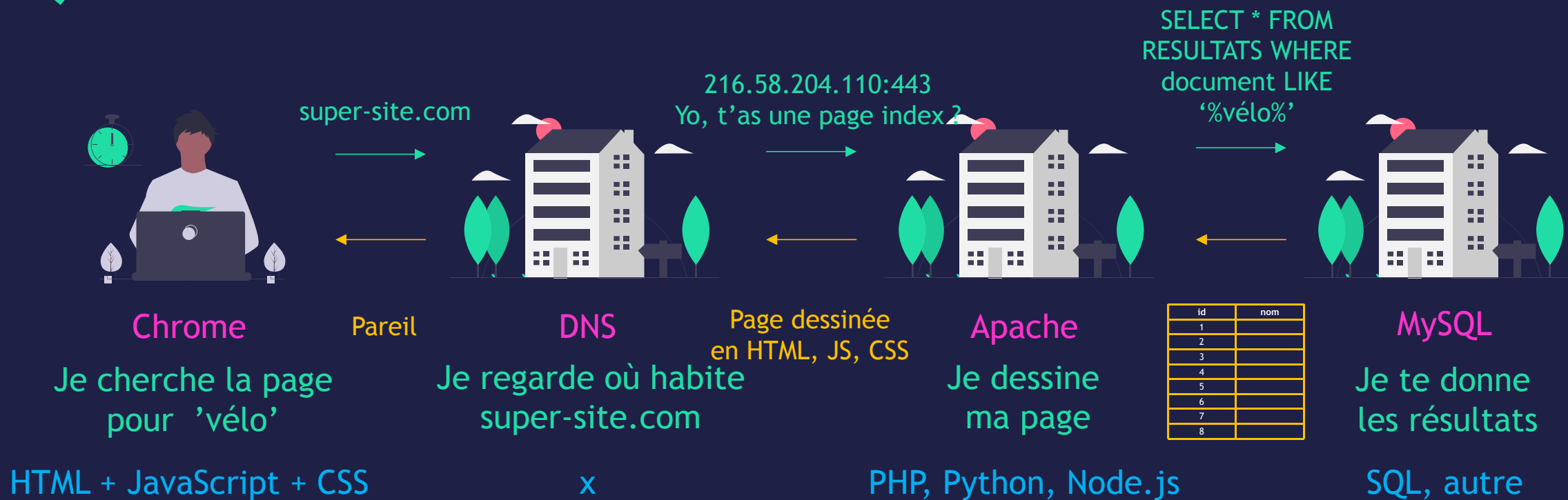




**YES!**











DNS



Chrome



Apache



MySQL



COOKIE

SESSION



COOKIE

SESSION



COOKIE

SESSION



COOKIE

SESSION





COOKIE

SESSION





# I. L'INFRASTRUCTURE

Qu'est-ce qu'un **réseau** ?



# Un serveur

# Un serveur

Une machine ou un logiciel fournissant  
des services à d'autres machines ou  
logiciels, par exemple 24h/24

# Un serveur

Une machine ou un logiciel fournissant  
des services à d'autres machines ou  
logiciels, par exemple 24h/24







Qu'est-ce que **WEB** ?

Une manière de partager de  
l'information sur internet

# Une manière de partager de l'information sur internet



## II. LE TRANSPORT

Transport l'information entre  
les moyens matériels de  
manière sécurisée

# Architecture client-serveur

1. Un client fait une requête à un serveur
2. Le serveur donne une réponse au client



DNS

Je regarde où habite  
google



Chrome

Je cherche la page  
de google pour 'vélo'

HTML + JavaScript + CSS

216.58.204.110:443  
Yo, t'as une page index ?



Apache

Je dessine  
ma page

PHP, Python, Node.js

SELECT \* FROM  
RESULTATS WHERE  
document LIKE  
'%vélo%'



MySQL

Je te donne  
les résultats

SQL, autre

id	nom
1	
2	
3	
4	
5	
6	
7	
8	



216.58.204.110:443  
Yo, t'as une page index ?



Chrome

Je cherche la page  
de google pour 'vélo'

HTML + JavaScript + CSS

Page dessinée  
en HTML, JS, CSS



Apache

Je dessine  
ma page

PHP, Python, Node.js

SELECT \* FROM  
RESULTATS WHERE  
document LIKE  
'%vélo%'



MySQL

Je te donne  
les résultats

SQL, autre

id	nom
1	
2	
3	
4	
5	
6	
7	
8	

# LE TRANSPORT BACKEND-FRONTEND

216.58.204.110:443  
Yo, t'as une page index ?



Chrome

Je cherche la page  
de google pour 'vélo'

HTML + JavaScript + CSS



Apache

Je dessine  
ma page

PHP, Python, Node.js

Page dessinée  
en HTML, JS, CSS

4xx

# Requête HTTP - GET, POST

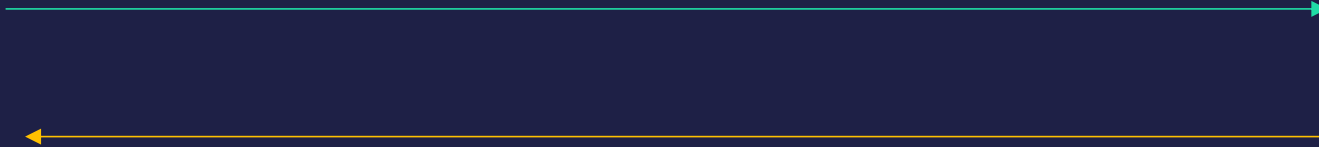
5xx



Chrome

Je cherche la page  
de google pour 'vélo'

HTML + JavaScript + CSS  
EXECUTION DE LA REPONSE



REPONSE HTML, CSS, JS +  
CODE



Apache

Je dessine  
ma page

PHP, Python, Node.js  
EXUCUTION DU PGM

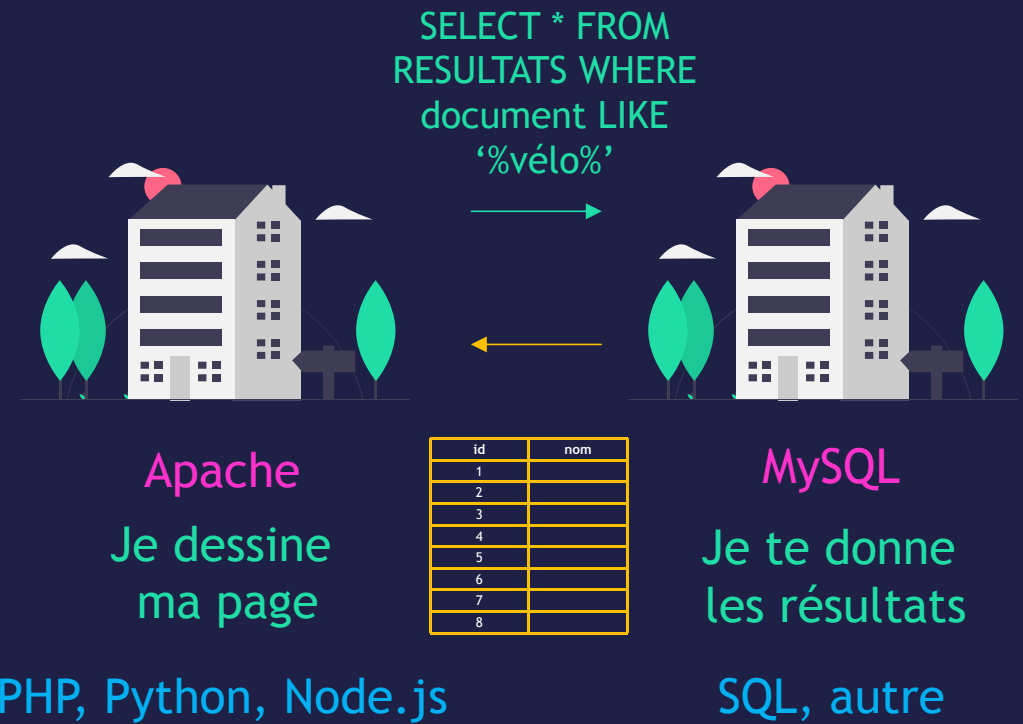
2xx

3xx

4xx

5xx

# LE TRANSPORT BACKEND-BACKEND



# Requête SQL

```
SELECT * FROM  
RESULTATS WHERE  
document LIKE  
'%vélo%'
```



Apache

Je dessine  
ma page

PHP, Python, Node.js



MySQL

Je te donne  
les résultats

SQL, autre

id	nom
1	
2	
3	
4	
5	
6	
7	
8	

Tableau

# III. L'APPLICATION



**Couper** son code en **3**  
**parties**  
**M, V et C**

MVC :  
Model  
View  
Controller

MVC :  
Model = Entités

MVC :

Model = Entités

View = Code pour le visuel

MVC :

Model = Entités

View = Code pour le visuel

Controller = Le reste

MVC :

Model = Entités

View = Code pour le visuel

Controller = Le reste

# Architecture MVC

