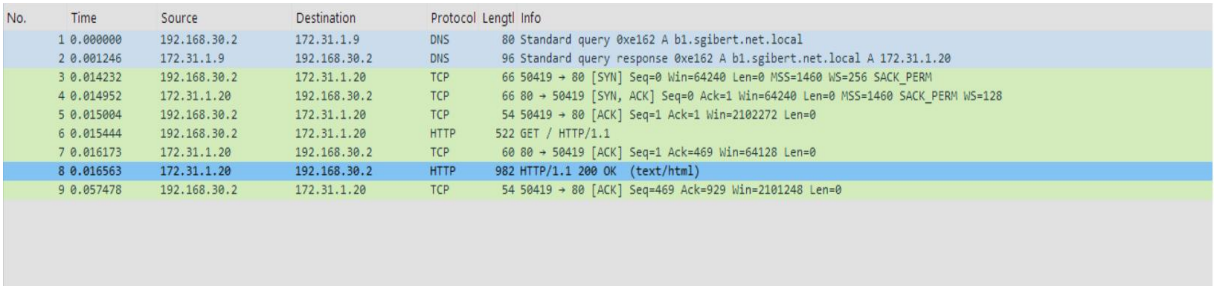
CENGIZ YASIN
HAMURCU ERENAY

TP1-WEB

Pour commencer avec Wireshark j'ouvre donc le document TrameHTTP.pcap pour voir la trame



No.	Time	Source	Destination	Protocol	Lengtl	Info
1	0.000000	192.168.30.2	172.31.1.9	DNS	80	Standard query 0xe162 A b1.sgibert.net.local
2	0.001246	172.31.1.9	192.168.30.2	DNS	96	Standard query response 0xe162 A b1.sgibert.net.local A 172.31.1.20
3	0.014232	192.168.30.2	172.31.1.20	TCP	66	50419 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
4	0.014952	172.31.1.20	192.168.30.2	TCP	66	80 → 50419 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_PERM WS=128
5	0.015004	192.168.30.2	172.31.1.20	TCP	54	50419 → 80 [ACK] Seq=1 Ack=1 Win=2102272 Len=0
6	0.015444	192.168.30.2	172.31.1.20	HTTP	522	GET / HTTP/1.1
7	0.016173	172.31.1.20	192.168.30.2	TCP	60	80 → 50419 [ACK] Seq=1 Ack=469 Win=64128 Len=0
8	0.016563	172.31.1.20	192.168.30.2	HTTP	982	HTTP/1.1 200 OK (text/html)
9	0.057478	192.168.30.2	172.31.1.20	TCP	54	50419 → 80 [ACK] Seq=469 Ack=929 Win=2101248 Len=0

Après on a ses protocoles : OSI : FTP, TCP, UDP, Telnet, HTTP, IP, Ethernet, DNS que l'on doit assigner a des couches du modèle OSI.

Couche 7 (Application) : HTTP, FTP, DNS, Telnet.

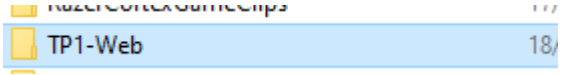
Couche 4 (Transport) : TCP, UDP.

Couche 3 (Réseau) : IP.

Couche 2 (Liaison) : Ethernet.

LES DIFFERENTS MODES

J'ai donc créé le dossier tP1-Web

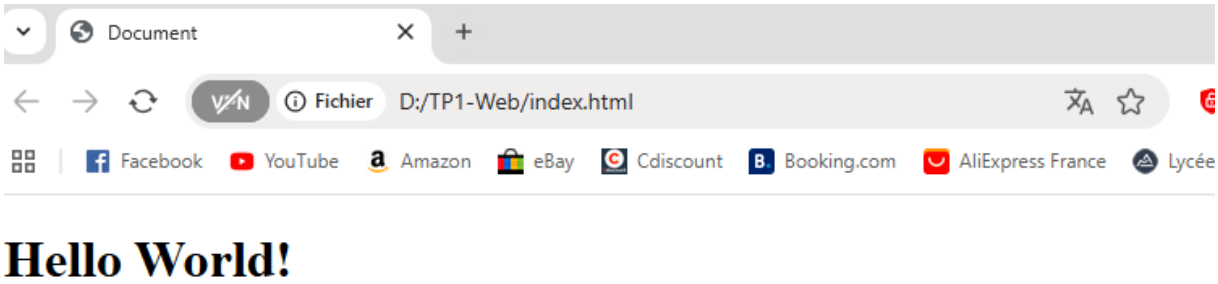


Je l'ai ouvert avec VScode, créer le fichier index.html et ajouter du code

```
index.html > html
1   <!DOCTYPE html>
2   <html lang="en">
3   <head>
4       <meta charset="UTF-8">
5       <meta name="viewport" content="width=device-width, initial-scale=1.0">
6       <title>Document</title>
7   </head>
8   <body>
9       <h1>Hello World!</h1>
10  </body>
11  </html>
```

Voilà ce que j'ai obtenu avec le code quand j'ai écrit start index.html sur le terminal.

CENGIZ YASIN
HAMURCU ERENAY



La page web s'ouvre bien et affiche donc Hello world ! comme prévu.

Suite à cela j'ai du installer python pour pouvoir faire ce code suivant :

python -m http.server

```
PS D:\TP1-Web> python -m http.server
Serving HTTP on :: port 8000 (http://[::]:8000/) ...
```

et donc cela à permis donc de pouvoir faire en sorte qu'on puisse accéder à ce site donc si j'accède à l'url depuis le terminal avec ce code : start chrome http://localhost:8000 cela m'ouvre google avec la page « Hello world ! » créé précédemment.

```
Serving HTTP on :: port 8000 (http://[::]:8000/) ...
::1 - - [22/Jan/2026 16:23:36] "GET / HTTP/1.1" 200 -
::1 - - [22/Jan/2026 16:23:38] code 404, message File not found
::1 - - [22/Jan/2026 16:23:38] "GET /favicon.ico HTTP/1.1" 404 -
```

Ce texte est apparu suite à mon accès au site et indique que j'ai visité la page à tel heure.

Avec mon binôme on doit réussir à pouvoir accéder au site de l'autre donc mon binôme devra accéder à mon site suite à ces étapes :

On remet ce code sur l'un des pc : python -m http.server

Pour avoir accès au site et depuis l'autre pc qui doit être connecté sur le même réseau pour fonctionner on doit taper : http://@IP_SERVEUR_DU_BINOME:8000 e l'occurrence ici l'url est http://172.31.1.85 qui correspond à mon adresse ipv4 que j'ai obtenu après avoir tapé ipconfig sur

CENGIZ YASIN
HAMURCU ERENAY

l’invite de commande.

```
arte Ethernet Ethernet 3 :

  Suffixe DNS propre à la connexion. . . : sioal.local
  Adresse IPv6 de liaison locale. . . . .: fe80::61f3:4df:7e87:d343%
  Adresse IPv4. . . . . . . . . . . . . .: 172.31.1.85
  Masque de sous-réseau. . . . . . . . . : 255.255.255.0
  Passerelle par défaut. . . . . . . . . : 172.31.1.254

arte réseau sans fil Connexion au réseau local* 18 :

  Statut du média. . . . . . . . . . . . : Média déconnecté
  Suffixe DNS propre à la connexion. . . :

arte réseau sans fil Connexion au réseau local* 19 :
```

Et cela a marcher il est bien arrivé sur le site.

à l’aide de la commande suivante j’ai réussi à repérer les ports en écoute (LISTENNING) et les ports connectés (ESTABLISHED) netstat -a -n

```
TCP    172.31.1.85:60506      20.189.173.17:443      ESTABLISHED
TCP    [::]:135               [::]:0                 LISTENING
TCP    [::]:445               [::]:0                 LISTENING
TCP    [::]:5357              [::]:0                 LISTENING
TCP    [::]:8000              [::]:0                 LISTENING
TCP    [::]:49664             [::]:0                 LISTENING
TCP    [::]:49665             [::]:0                 LISTENING
TCP    [::]:49666             [::]:0                 LISTENING
TCP    [::]:49667             [::]:0                 LISTENING
TCP    [::]:49672             [::]:0                 LISTENING
TCP    [::]:49673             [::]:0                 LISTENING
TCP    [::]:51038             [::]:0                 LISTENING
TCP    [::1]:42050            [::]:0                 LISTENING
TCP    [::1]:42050            [::1]:59365            ESTABLISHED
TCP    [::1]:49671            [::]:0                 LISTENING
TCP    [::1]:59365            [::1]:42050            ESTABLISHED
UDP    0.0.0.0:500            *:*
```

```
  Proto  Adresse locale         Adresse distante       État
  TCP    0.0.0.0:135            0.0.0.0:0              LISTENING
  TCP    0.0.0.0:445            0.0.0.0:0              LISTENING
  TCP    0.0.0.0:5040           0.0.0.0:0              LISTENING
  TCP    0.0.0.0:5357           0.0.0.0:0              LISTENING
  TCP    0.0.0.0:8000           0.0.0.0:0              LISTENING
  TCP    0.0.0.0:49664          0.0.0.0:0              LISTENING
  TCP    0.0.0.0:49665          0.0.0.0:0              LISTENING
  TCP    0.0.0.0:49666          0.0.0.0:0              LISTENING
  TCP    0.0.0.0:49667          0.0.0.0:0              LISTENING
  TCP    0.0.0.0:49672          0.0.0.0:0              LISTENING
  TCP    0.0.0.0:49673          0.0.0.0:0              LISTENING
  TCP    0.0.0.0:51038          0.0.0.0:0              LISTENING
  TCP    127.0.0.1:3369         0.0.0.0:0              LISTENING
  TCP    127.0.0.1:13333        0.0.0.0:0              LISTENING
  TCP    127.0.0.1:49673        127.0.0.1:63245        ESTABLISHED
  TCP    127.0.0.1:55000        0.0.0.0:0              LISTENING
  TCP    127.0.0.1:56975        0.0.0.0:0              LISTENING
  TCP    127.0.0.1:59214        0.0.0.0:0              LISTENING
  TCP    127.0.0.1:63245        127.0.0.1:49673        ESTABLISHED
  TCP    172.31.1.85:139        0.0.0.0:0              LISTENING
  TCP    172.31.1.85:59433      3.232.144.130:443      ESTABLISHED
  TCP    172.31.1.85:59435      74.125.206.188:5228    ESTABLISHED
  TCP    172.31.1.85:59439      98.66.133.186:443      ESTABLISHED
  TCP    172.31.1.85:60029      216.58.213.74:443      ESTABLISHED
```

CENGIZ YASIN
HAMURCU ERENAY

On peut apercevoir aussi l'état des connexions TCP/UDP. Elles confirment que le port 8000 est bien en écoute (LISTENING sur TCP 0.0.0.0:8000 et [::]:8000), ce qui correspond au serveur Python lancé.

Maintenant je dois trouver des informations tel que :

Quel est le nom de la méthode ? Son URL ? La version du protocole http utilisé

Les informations identifiées dans le terminal :

Méthode de la requête : GET

Indiqué sur la ligne COMMENTAIRES : GET with 0-byte payload.

URL : http://172.31.1.85:8000

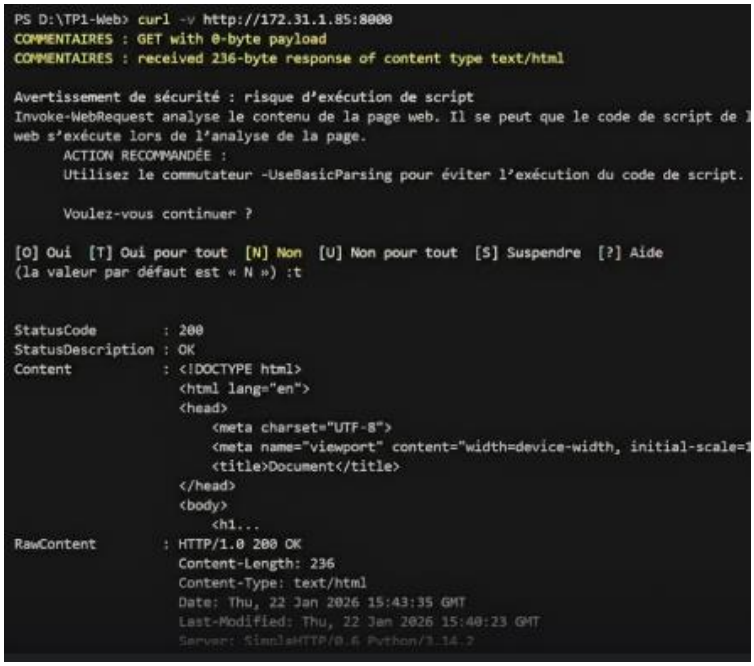
C'est l'adresse saisie dans la ligne de commande

Version du protocole HTTP : HTTP/1.0

Cette information apparaît dans la section :

RawContent : HTTP/1.0 200 OK.

Ici on va tester la commande curl -v http://172.31.1.85:8000 : pour voir en détail une requête http



Ici on peut donc voir détail comment est un requête http.

CENGIZ YASIN
HAMURCU ERENAY

D’après la capture d'écran, on voit clairement StatusCode : 200 et StatusDescription : OK.

> Explication : C'est le code de succès standard en HTTP. Il signifie que le serveur a bien reçu la requête du client, qu'il a trouvé la ressource demandée et qu'il l'a renvoyée sans problème.

Si on voit une erreur 404 dans le terminal, c'est que le dialogue a échoué sur la fin.

> Explication de l'erreur : Le code 404 signifie que le serveur communique bien avec le client, mais qu'il ne trouve absolument rien à l'adresse (URL) demandée.
>
> Cela peut arriver si on fait une faute de frappe ou si le fichier a été déplacé ou re-nommé

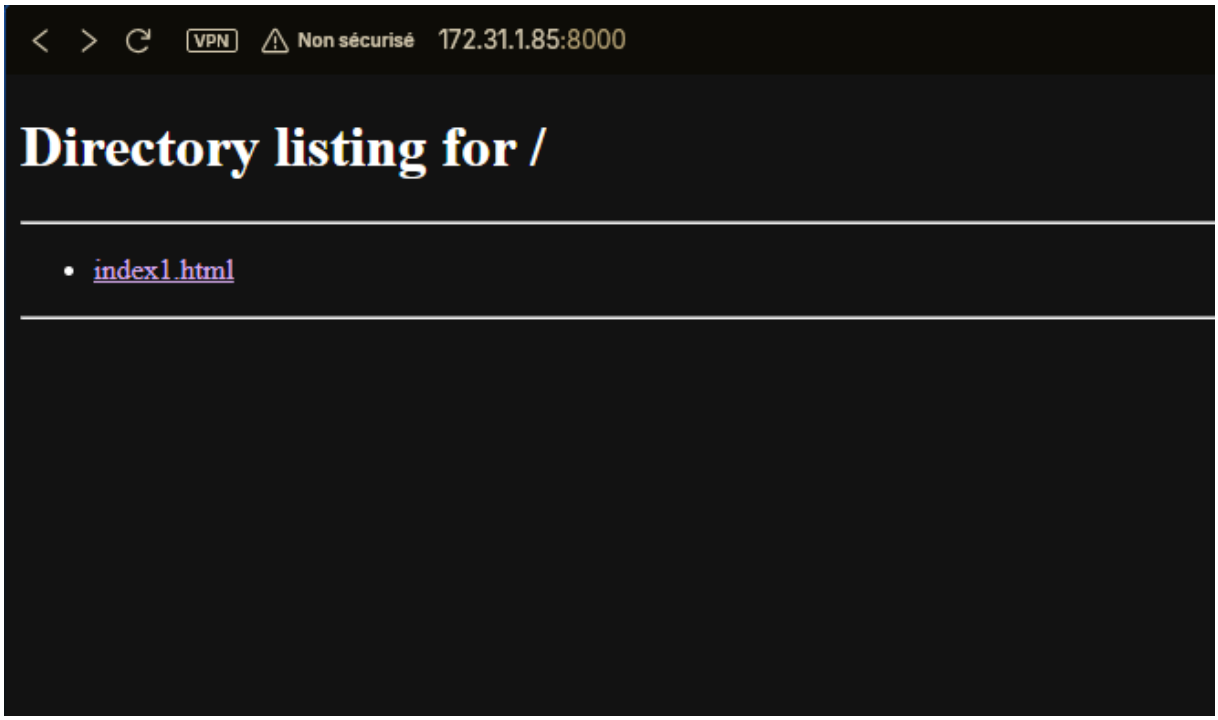
Pour que le serveur renvoie à nouveau un code 200 (Succès), j'ai deux options :

> Côté Client : Je vérifie l'URL tapée dans mon navigateur. Je m'assure que le nom du fichier et l'extension (.html) sont parfaitement identiques à ce qui est stocké sur le disque.
>
> Côté Serveur : Je vérifie que mon fichier est bien placé dans le dossier où j'ai lancé la commande python -m http.server. Si le fichier est dans un sous-dossier, je dois l'inclure dans l'URL (ex: /dossier/page.html).

ETUDE DES FONCTIONNALITES D’UN SERVEUR WEB

J'ai renommé le fichier index.html en index1.html.

Le constat : Quand j'actualise ma page sur http://172.31.1.85:8000/, le site disparaît. À la place, je me retrouve face à une liste de fichiers et le "Directory Listing", je vois mon fichier index1.html qui m'attend dans la liste, on se demande donc pourquoi ?

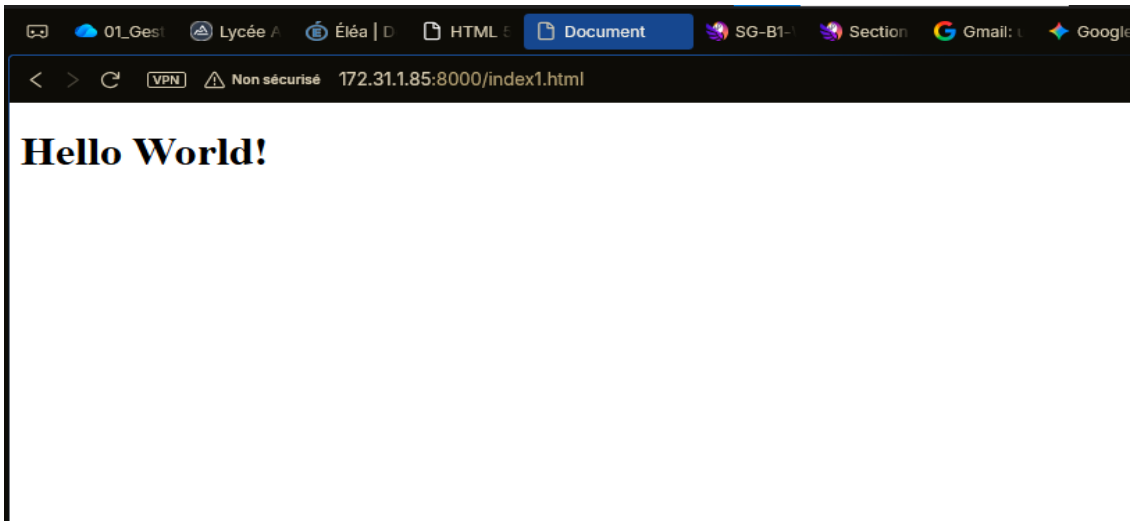


CENGIZ YASIN
HAMURCU ERENAY

Cela est dû au serveur HTTP de Python qui est programmé pour être un bon petit soldat. Quand je lui demande l'adresse racine, sa première mission est de chercher un fichier nommé exactement index.html. Comme je l'ai renommé, il ne le trouve plus. Il se dit : puisque la page d'accueil officielle n'est plus là, je vais simplement montrer à l'utilisateur tout ce qu'il y a dans le dossier.

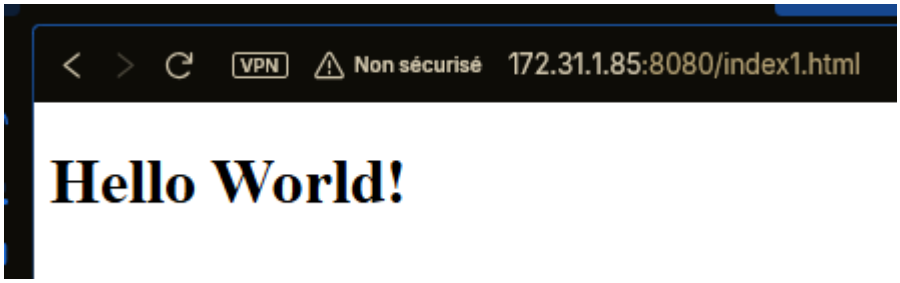
Pour récupérer ma page, je ne peux plus compter sur l'automatisme du serveur. Je dois être précis et lui dire exactement ce que je veux. Mon nouveau URL : Je dois ajouter le nom exact du fichier à la fin de l'adresse

http://172.31.1.85:8000/index1.html



Si je relance le serveur avec la commande python -m http.server 8080, le serveur n'écoute plus sur son canal habituel.

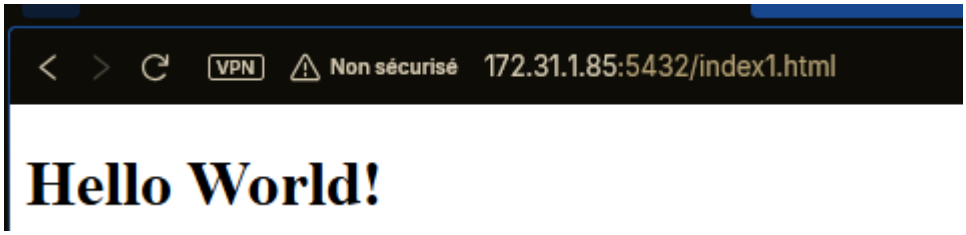
> Pour que la page s'affiche : Je dois impérativement modifier l'URL dans mon navigateur. Je dois maintenant taper : http://172.31.1.85:8080/index1.html



Si on choisit le port 5432 :

Est-ce que ça fonctionne ? Oui, cela fonctionnera toujours si je précise bien :5432 dans l'URL.

CENGIZ YASIN
HAMURCU ERENAY



Pourquoi ?

Un serveur peut écouter sur n'importe quel numéro de port libre entre 1 et 65535. Le port 5432 est techniquement un port comme un autre. Tant qu'aucun autre service n'utilise ce port sur ma machine, Python peut s'y installer.

Lancer deux serveurs en même temps est-ce que ça marche ?

Oui, techniquement c'est tout à fait possible, à condition d'utiliser deux numéros de ports différents (par exemple un sur le 8000 et l'autre sur le 8080).

Quel intérêt ? Je peux héberger deux sites ou deux versions d'une application différentes sur la même machine mais on peut aussi utiliser pour beaucoup d'autre chose.

J'ai fini par mettre le port a 8000 par défaut

J'ai créé un fichier test.css et j'ai ajouté ce code

```
body{
background-color: #f3b8b8;
}
```

Ensuite j'ai essayé la commande curl -v http://localhost:8000/test.css

CENGIZ YASIN
HAMURCU ERENAY

```
PS D:\TP1-Web> curl -v http://localhost:8000/test.css
COMMENTAIRES : GET http://localhost:8000/test.css with 0-byte payload
COMMENTAIRES : received 0-byte response of content type text/css


StatusCode        : 200
StatusDescription : OK
Content           :
RawContent        : HTTP/1.0 200 OK
                    Content-Length: 0
                    Content-Type: text/css
                    Date: Thu, 22 Jan 2026 17:41:59 GMT
                    Last-Modified: Thu, 22 Jan 2026 17:40:50 GMT
                    Server: SimpleHTTP/0.6 Python/3.14.2


Forms             : {}
Headers           : {[Content-Length, 0], [Content-Type, text/css], [Date, Thu, 22 Jan 2026 17:41:59 GMT],
                    [Last-Modified, Thu, 22 Jan 2026 17:40:50 GMT]...}
Images            : {}
InputFields       : {}
Links             : {}
ParsedHtml        : mshtml.HTMLDocumentClass
RawContentLength  : 0



PS D:\TP1-Web>
```

Ici on voit que le « content type » est text/css

```
Serving HTTP on :: port 8000 (http://[::]:8000/) ...
::1 - - [08/Jan/2026 09:17:04] "GET /test.css HTTP/1.1" 200 -
::1 - - [08/Jan/2026 09:18:32] "GET /index1.html HTTP/1.1" 304 -
::1 - - [08/Jan/2026 09:19:23] code 404, message File not found
::1 - - [08/Jan/2026 09:19:23] "GET /index1.html HTTP/1.1" 404 -
::1 - - [08/Jan/2026 09:19:30] "GET /index.html HTTP/1.1" 200 -
::1 - - [08/Jan/2026 09:19:32] "GET /index.html HTTP/1.1" 304 -
::1 - - [08/Jan/2026 09:19:59] "GET /index.html HTTP/1.1" 200 -
::1 - - [08/Jan/2026 09:19:59] "GET /test.css HTTP/1.1" 200 -
::1 - - [08/Jan/2026 09:24:18] "GET /a.png HTTP/1.1" 200 -
::1 - - [08/Jan/2026 09:24:36] "GET /a.docx HTTP/1.1" 200
```

Quand je regarde les logs, je peux identifier plusieurs scénarios :

Le code 304 : Cela signifie "Non modifié". Le serveur dit au client qu'il as déjà ce fichier en mémoire.

Le code 404 : On voit bien la trace d'erreur quand la recherche index1.html a été faites alors qu'il n'existait pas ou plus : code 404, message File not found.