

**Mise en place des applications Web**

Projet gsb.org

DUMAS Lucie

Table des matières

[Qu’est-ce qu’Nginx 3](#_Toc167369589)

[Nginx 3](#_Toc167369590)

[Nginx ou Apache ? 3](#_Toc167369591)

[Création des serveurs web 5](#_Toc167369592)

[DokuWiki 6](#_Toc167369593)

[Wordpress 10](#_Toc167369594)

[Configuration de la base de données 10](#_Toc167369595)

[Installation et configuration de Wordpress 11](#_Toc167369596)

[.htaccess 17](#_Toc167369597)

# Qu’est-ce qu’Nginx

## Nginx

Nginx (prononcé “engine-x”) est un serveur web open source de haute performance, ainsi qu'un serveur proxy inverse. Il a été conçu pour être léger, rapide et capable de gérer un grand nombre de connexions simultanées. Initialement créé par Igor Sysoev en 2004, Nginx est aujourd'hui largement utilisé pour servir des sites web et des applications web, en concurrence avec d'autres serveurs web tels qu'Apache.

Nginx est un serveur web populaire en raison de sa rapidité, de sa légèreté, de sa flexibilité et de sa capacité à gérer efficacement les charges de trafic importantes. Il est souvent utilisé comme composant essentiel dans l'infrastructure web pour améliorer les performances et la disponibilité des applications en ligne.

## Nginx ou Apache ?

Le choix entre Nginx et Apache dépend largement des besoins spécifiques de l'application ou du site web, ainsi que des caractéristiques de performance, de configuration et de fonctionnalités recherchées.

Utiliser Nginx si :

* Haute performance : Nginx excelle dans la gestion d'un grand nombre de connexions simultanées et est réputé pour ses performances élevées, ce qui en fait un bon choix pour les sites web à fort trafic.
* Serveur proxy inverse : Nginx est souvent privilégié pour créer un serveur proxy inverse pour équilibrer la charge entre plusieurs serveurs en raison de ses performances et de son efficacité.
* Contenu statique : Si le site web délivre principalement du contenu statique (images, CSS, JavaScript), Nginx est particulièrement efficace dans ce domaine.
* Réécriture d'URL : Nginx offre une syntaxe claire et puissante pour les fonctionnalités avancées de réécriture d'URL pour personnaliser les liens et les redirections.
* Configuration simple : Nginx a une configuration simple et lisible, ce qui facilite la gestion et la compréhension des fichiers de configuration.

Utiliser Apache si :

* Compatibilité avec les modules : Si l'application nécessite des modules spécifiques qui sont mieux pris en charge par Apache, il peut être préférable de choisir Apache. Certains modules peuvent ne pas être disponibles ou avoir une prise en charge limitée dans Nginx.
* Hébergement partagé : Apache est souvent utilisé dans des environnements d'hébergement partagé en raison de sa flexibilité et de sa facilité d'utilisation pour les configurations virtuelles.
* .htaccess : Apache offre un meilleur support pour l'utilisation des fichiers .htaccess pour la configuration dynamique au niveau du répertoire.

Le choix entre Nginx et Apache dépend donc de la nature spécifique des projet, préférences personnelles, compétences et fonctionnalités requises. Certains sites web utilisent également les deux serveurs en tandem, avec Nginx en amont (en tant que proxy inverse) et Apache en aval pour traiter les requêtes. Cela dépendra de la configuration et des exigences spécifiques des infrastructure web.

# Création des serveurs web

Nous clonons dans un premier temps la machine virtuelle template et lui attribuons une nouvelle adresse IP (10.31.177.80 (srv-web1-1) ou 10.31.178.80 (srv-web2-1)) et un nouveau nom d'hôte dans le DHCP :

dhclient -r && dhclient -v

# Change le nom d'hôte à l'aide d'une commande

hostnamectl set-hostname srv-web1-1 # Ou srv-web1-2

# Change le nom d'hôte directement depuis le fichier de configuration

nano /etc/hosts

Nous modifions dans le fichier /etc/hosts le nom d’hôte de la machine et nous nous déconnectons pour actualiser l’affichage du nom.

Nous installons les paquets nécessaires :

apt update && apt upgrade

apt install nginx

Nous pouvons maintenant cloner le serveur web pour et lui attribuons une nouvelle adresse IP (10.31.185.80 (srv-web1-2) ou 10.31.185.80 (srv-web2-2)) et un nouveau nom d'hôte dans le DHCP :

dhclient -r && dhclient -v

# Change le nom d'hôte à l'aide d'une commande

hostnamectl set-hostname srv-web2-1 # Ou srv-web2-2

# Change le nom d'hôte directement depuis le fichier de configuration

nano /etc/hosts

# DokuWiki

Notre site de documentation sera situé sur notre serveur web privé, de même que notre serveur NextCloud, que nous mettrons en place lors de notre prochaine mission dans laquelle nous chiffrerons nos communications. En effet, la configuration de NextCloud est adaptée pour le protocole HTTPS et très complexe à adapter pour le protocole HTTP.

Nous commençons donc par créer le dossier qui accueillera le DokuWiki :

mkdir -p /home/htdocs/gsb.org/wiki/

Nous devons également installer les paquets liés à PHP qui permettront au Wiki de fonctionner, ainsi que les paquets Nginx :

apt install php php-xml php-fpm nginx

Nous devons maintenant télécharger la dernière archive de DokuWiki :

wget https://download.dokuwiki.org/src/dokuwiki/dokuwiki-stable.tgz

Nous extrayons l’archive nouvellement téléchargée :

tar xf dokuwiki-stable.tgz

Nous vérifions la version de notre DokuWiki téléchargée à l’aide de la commande ls.



Renommons notre dossier :

mv dokuwiki-2023-04-04a wiki

Nous créons le fichier de configuration DokuWiki /etc/nginx/sites-available/documentation.asie.gsb.org :

server {

server\_name documentation.asie.gsb.org;

listen 80;

listen [::]:80;

autoindex off;

#client\_max\_body\_size 15M;

#client\_body\_buffer\_size 128k;

index index.html index.htm index.php doku.php;

root /home/htdocs/gsb.org/wiki;

location / {

try\_files $uri $uri/ @dokuwiki;

}

location ~ ^/lib.\*\.(gif|png|ico|jpg)$ {

expires 30d;

}

location = /robots.txt { access\_log off; log\_not\_found off; }

location = /favicon.ico { access\_log off; log\_not\_found off; }

location ~ /\. { access\_log off; log\_not\_found off; deny all; }

location ~ ~$ { access\_log off; log\_not\_found off; deny all; }

location @dokuwiki {

rewrite ^/\_media/(.\*) /lib/exe/fetch.php?media=$1 last;

rewrite ^/\_detail/(.\*) /lib/exe/detail.php?media=$1 last;

rewrite ^/\_export/([^/]+)/(.\*) /doku.php?do=export\_$1&id=$2 last;

rewrite ^/(.\*) /doku.php?id=$1 last;

}

location ~ \.php$ {

try\_files $uri =404;

fastcgi\_pass unix:/var/run/php/php8.2-fpm.sock;

fastcgi\_index index.php;

fastcgi\_param SCRIPT\_FILENAME $document\_root$fastcgi\_script\_name;

include /etc/nginx/fastcgi\_params;

fastcgi\_param QUERY\_STRING $query\_string;

fastcgi\_param REQUEST\_METHOD $request\_method;

fastcgi\_param CONTENT\_TYPE $content\_type;

fastcgi\_param CONTENT\_LENGTH $content\_length;

fastcgi\_intercept\_errors on;

fastcgi\_ignore\_client\_abort off;

fastcgi\_connect\_timeout 60;

fastcgi\_send\_timeout 180;

fastcgi\_read\_timeout 180;

fastcgi\_buffer\_size 128k;

fastcgi\_buffers 4 256k;

fastcgi\_busy\_buffers\_size 256k;

fastcgi\_temp\_file\_write\_size 256k;

}

location ~ /(data|conf|bin|inc)/ {

deny all;

}

location ~ /\.ht {

deny all;

}

}

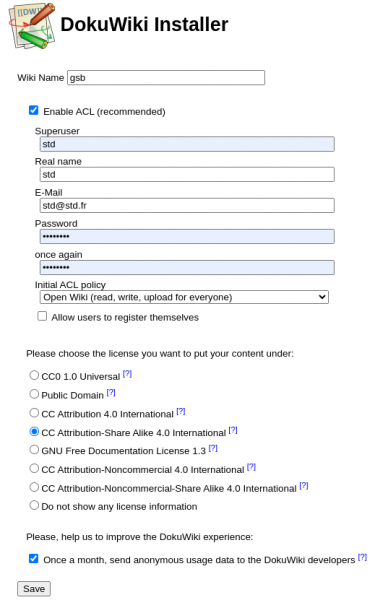
Nous créons maintenant notre lien symbolique permettant à Nginx d’activer le fichier de configuration lié :

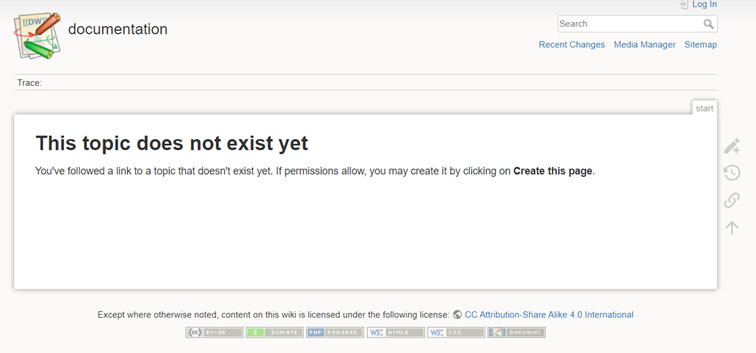
ln -s /etc/nginx/sites-available/documentation.asie.gsb.org /etc/nginx/sites-enable/documentation.asie.gsb.org

Nous redémarrons le service Nginx :

systemctl restart nginx

Nous pouvons dès à présent accéder à l’interface de configuration de DokuWiki à l’adresse documentation.asie.gsb.org :



 Après avoir configuré notre DokuWiki, nous pouvons accéder à notre page d’accueil (qui n’existe pas encore) :

# Wordpress

## Configuration de la base de données

Wordpress est un CMS nécessitant une base de données. Nous allons donc commencer par configurer cette basse de données. Sur notre serveur de bases de données, nous nous connectons à l’interface MariaDB :

mysql -u root -p

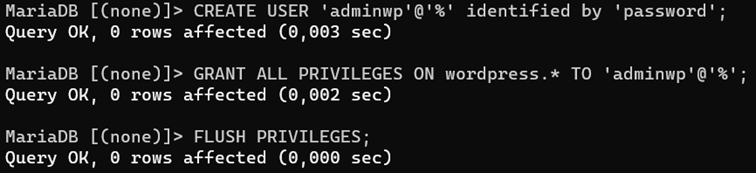
Nous devons créer deux bases de données possédant des comptes distincts. Nous créons la base de données liée au site www.gsb.org :

CREATE DATABASE wordpress;

CREATE USER 'adminwp'@'%' identified by 'password';

GRANT ALL PRIVILEGES ON wordpress.\* TO 'adminwp'@'%';

FLUSH PRIVILEGES;



Nous créons la base de données liée au site www.asie.gsb.org :

CREATE DATABASE wordpressasie;

CREATE USER 'adminwpasie'@'%' identified by 'password';

GRANT ALL PRIVILEGES ON wordpressasie.\* TO 'adminwpasie'@'%';

FLUSH PRIVILEGES;

Une image contenant texte, capture d’écran, Police

Description générée automatiquement

## Installation et configuration de Wordpress

Nos sites web Wordpress se situeront sur notre serveur web public. Nous commençons par télécharger les paquets nécessaires pour le fonctionnement de Wordpress :

apt update && apt upgrade

apt install php-fpm

Nous devons à présent créer les répertoires qui accueilleront nos nouveaux sites internet :

# Création du dossier accueillant le site www.asie.gsb.org

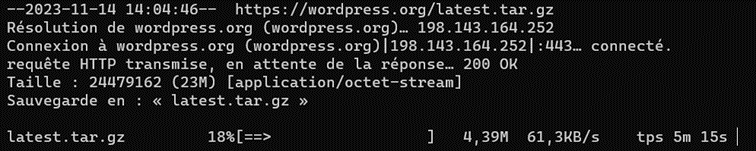
mkdir -p /home/htdocs/gsb.org/asie/

# Création du dossier accueillant le site www.gsb.org

mkdir -p /home/htdocs/gsb.org/gsb/

Nous nous déplaçons dans le dossier dans lequel nous voulons créer notre site internet et téléchargeons la dernière version de Wordpress :

wget https://wordpress.org/latest.tar. gz



Nous extrayons l’archive Wordpress téléchargée précédemment :

tar -xvzf latest.tar.gz

Nous nous déplaçons dans le dossier créé nommé wordpress et changeons le nom du fichier exemple de configuration :

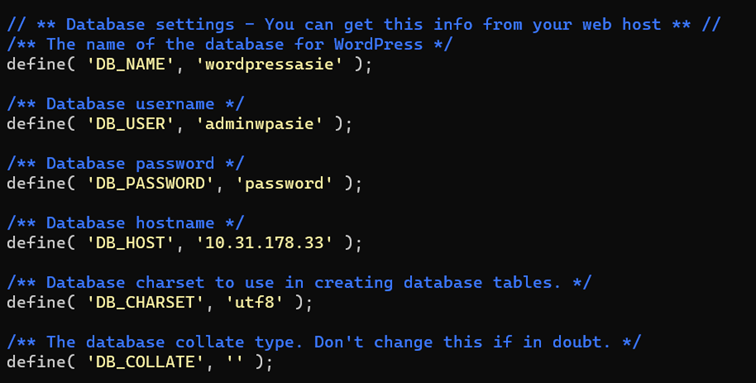
cd wordpress

cp wp-config-sample.php wp-config.php

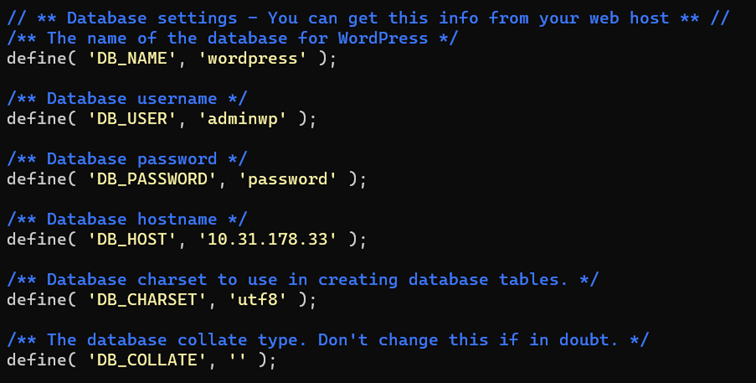
Nous devons maintenant modifier des éléments de notre fichier de configuration. Pour ce faire, nous modifions le fichier wp-config.php :

nano wp-config.php

Nous devons rajouter dans les endroits prévus à cet effet le nom de notre base de données créée précédemment, le nom de l’utilisateur de cette base de données, son mot de passe ainsi que l’adresse IP de la machine sur laquelle est stockée la base de données.



fichier de configuration /home/htdocs/gsb.org/asie/wordpress/wp-config.php



fichier de configuration /home/htdocs/gsb.org/gsb/wordpress/wp-config.php

A présent, nous devons créer les fichiers de configuration Nginx et activer notre site internet. Commençons tout d’abord par créer les fichiers /etc/nginx/sites-available/www.asie.gsb.org.conf et /etc/nginx/sites-available/asie.gsb.org.conf :

server {

server\_name www.asie.gsb.org; #ou www.gsb.org; selon le site à configurer

listen 80;

listen [::]:80;

autoindex off;

#client\_max\_body\_size 15M;

#client\_body\_buffer\_size 128k;

index index.html index.htm index.php doku.php;

root /home/htdocs/gsb.org/asie/;

location / {

try\_files $uri $uri/ @dokuwiki;

}

location ~ ^/lib.\*\.(gif|png|ico|jpg)$ {

expires 30d;

}

location = /robots.txt { access\_log off; log\_not\_found off; }

location = /favicon.ico { access\_log off; log\_not\_found off; }Une image contenant texte, capture d’écran, Police

Description générée automatiquement

location ~ /\. { access\_log off; log\_not\_found off; deny all; }

location ~ ~$ { access\_log off; log\_not\_found off; deny all; }

location @dokuwiki {

rewrite ^/\_media/(.\*) /lib/exe/fetch.php?media=$1 last;

rewrite ^/\_detail/(.\*) /lib/exe/detail.php?media=$1 last;

rewrite ^/\_export/([^/]+)/(.\*) /doku.php?do=export\_$1&id=$2 last;

rewrite ^/(.\*) /doku.php?id=$1 last;

}

location ~ \.php$ {

try\_files $uri =404;

fastcgi\_pass unix:/var/run/php/php8.2-fpm.sock;

fastcgi\_index index.php;

fastcgi\_param SCRIPT\_FILENAME $document\_root$fastcgi\_script\_name;

include /etc/nginx/fastcgi\_params;

fastcgi\_param QUERY\_STRING $query\_string;

fastcgi\_param REQUEST\_METHOD $request\_method;

fastcgi\_param CONTENT\_TYPE $content\_type;

fastcgi\_param CONTENT\_LENGTH $content\_length;

fastcgi\_intercept\_errors on;

fastcgi\_ignore\_client\_abort off;

fastcgi\_connect\_timeout 60;

fastcgi\_send\_timeout 180;

fastcgi\_read\_timeout 180;

fastcgi\_buffer\_size 128k;

fastcgi\_buffers 4 256k;

fastcgi\_busy\_buffers\_size 256k;

fastcgi\_temp\_file\_write\_size 256k;

}

location ~ /(data|conf|bin|inc)/ {

deny all;

}

location ~ /\.ht {

deny all;

}

}

Nous créons maintenant un lien symbolique entre ce fichier de configuration et le dossier /etc/nginx/sites-enabled/ :

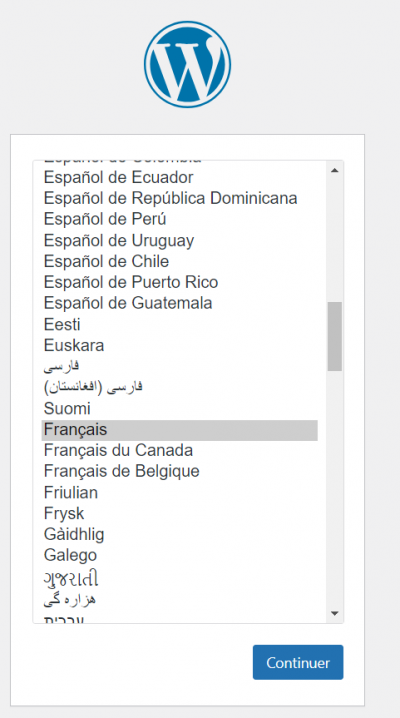
ln -s /etc/nginx/sites-available/www.asie.gsb.org.conf /etc/nginx/sites-enable/www.asie.gsb.org.conf

ln -s /etc/nginx/sites-available/www.gsb.org.conf /etc/nginx/sites-enable/www.gsb.org.conf

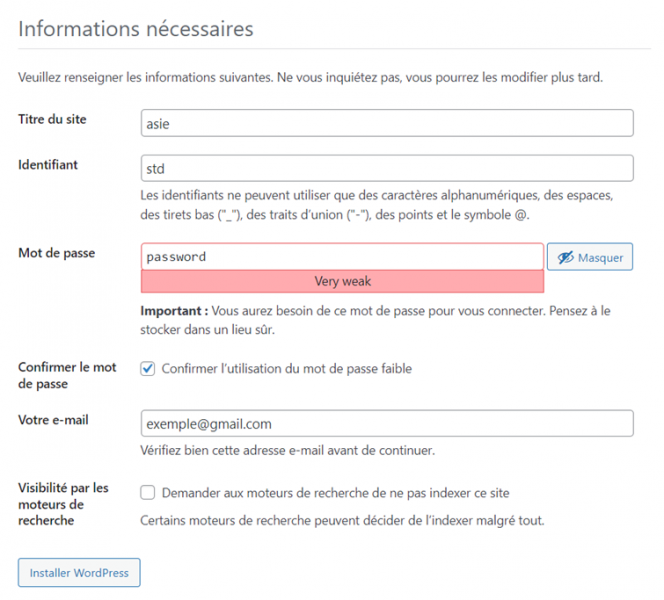
Nous redémarrons notre service afin que les changements soient effectifs :

systemctl restart nginx

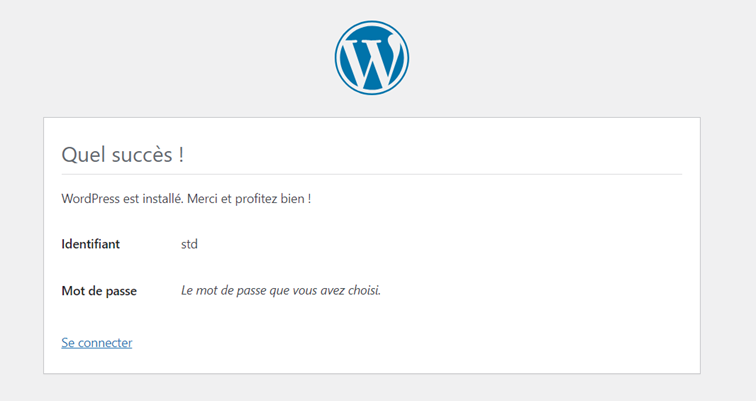
Nous pouvons maintenant accéder à l’interface graphique de Wordpress via nos adresses web www.asie.gsb.org et www.gsb.org. Nous commençons par sélectionner la langue de l’interface.

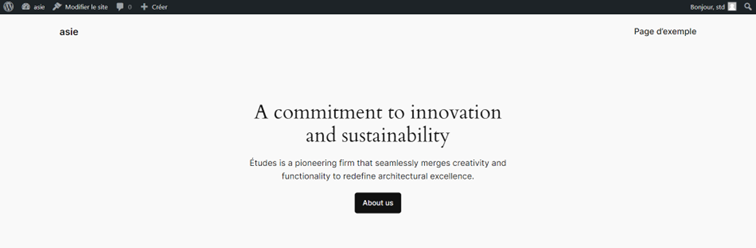


Nous configurons maintenant notre Wordpress en entrant le titre du site, l’identifiant de l’utilisateur, son mot de passe ainsi que son e-mail. Nous devons également cocher l’option « Confirmer l’utilisation du mot de passe faible » étant donné que notre mot de passe « password » n’est pas sécurisé.

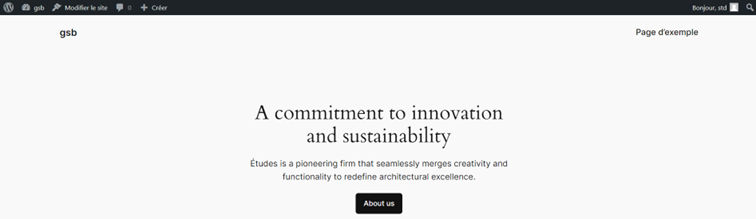


Notre site Wordpress est maintenant configuré.





Nous appliquons ces mêmes étapes pour configurer notre deuxième site Wordpress.



## .htaccess

Nous devons mettre en place une interface de connexion pour les espaces utilisateurs. Nous commençons par créer le dossier /home/std/public\_html qui servira de dossier utilisateur accessible en ligne, et lui attribuons les droits nécessaires :

mkdir /home/std/public\_html

chmod a+x /home/std

Cependant, contrairement à Apache, nous pouvons configurer directement les différentes directives dans le fichier de configuration du Virtual Host. Nous insérons donc dans chaque fichier de configuration des sites un nouveau bloc « location » permettant de définir le dossier /home/std/public\_html :

location ~ ^/~(.+?)(/.\*)?$ {

alias /home/$1/public\_html$2;

index index.html index.htm;

autoindex on;

auth\_basic "Zone securisée - Authentification requise";

auth\_basic\_user\_file /etc/nginx/.htpasswd;

}

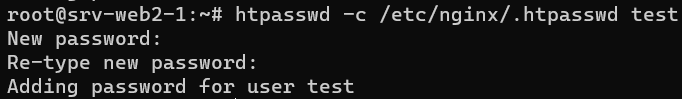
Nous devons maintenant créer un fichier htpasswd. Pour ce faire, nous commençons tout d’abord par télécharger le paquet apache2-utils :

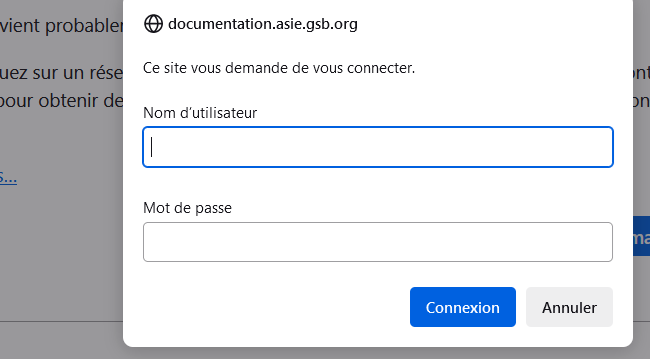
apt update && apt upgrade

apt install apache2-utils

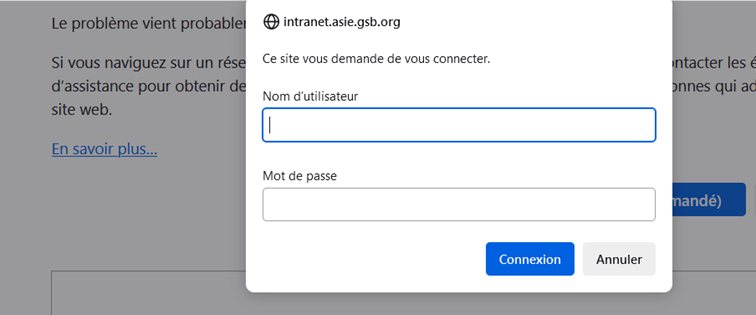
Le fichier htpasswd contient les identifiants et mots de passe des utilisateurs. Nous créons l’utilisateur « test » qui aura pour mot de passe « password » :

htpasswd -c /etc/nginx/.htpasswd test

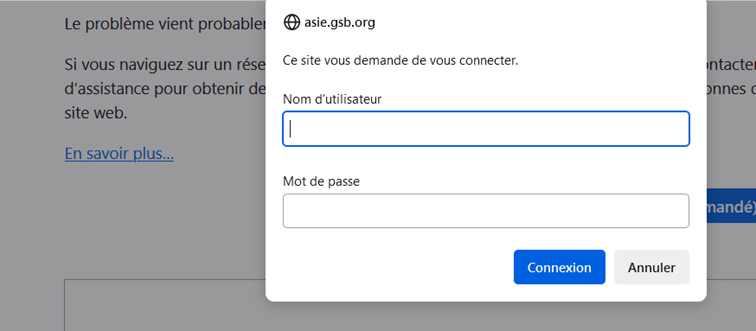


 Nous vérifions maintenant que chaque site est protégé :

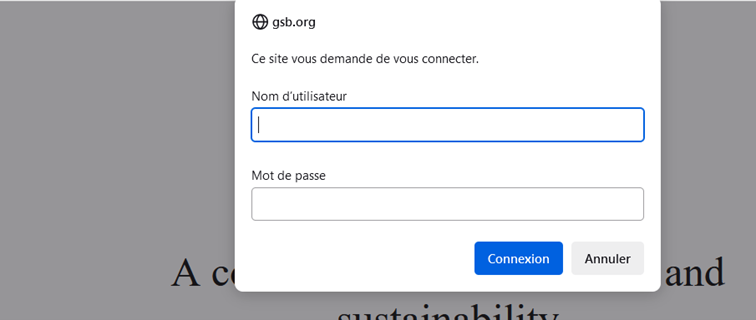
documentation.asie.gsb.org



intranet.asie.gsb.org



www.asie.gsb.org



www.gsb.org