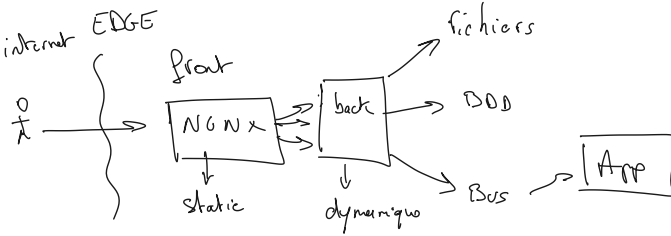
NGNIX

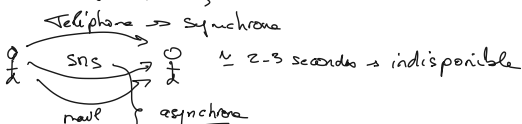
Archis :

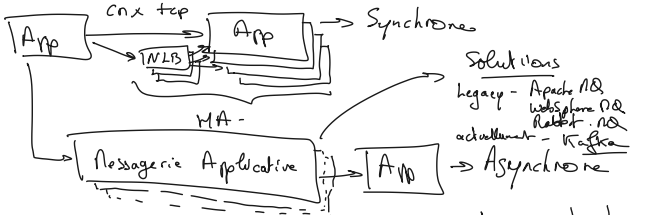


fichiers → EFS (NFS) fstab

BDD → MySql, RDS, MariaDB MA gérée par AWS

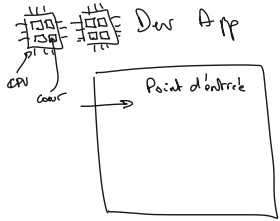
BUS → Rabbit MQ





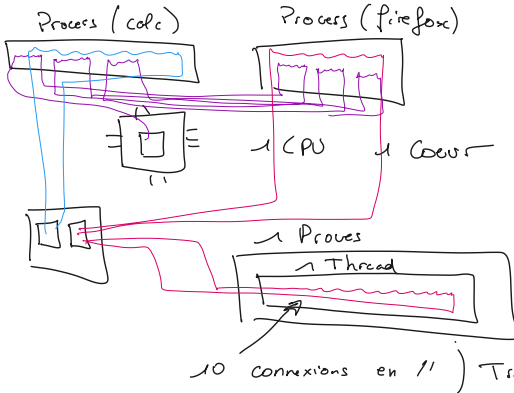
→ PUSH → je reçois l'information

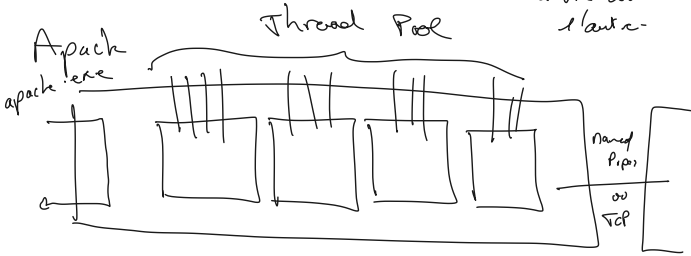
→ PULL → je vais chercher (scruter) l'information



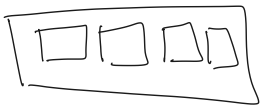
Multitache : → Thread (est géré par le process)
→ Process (est géré par l'OS)

à la base 1 Process = (Mémoire)
1 Thread (code)

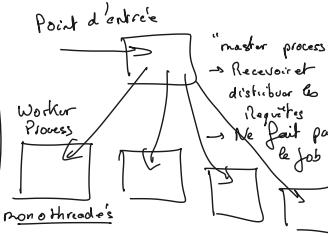




Apache = Mono Process
Multi thread



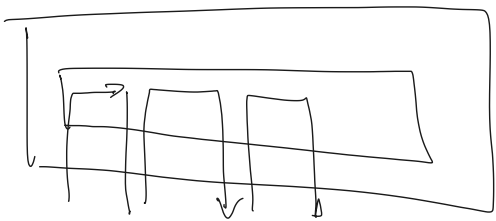
NGNX



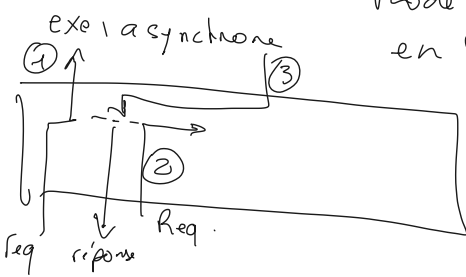
1 Process = 1 Thread = 1 coeur

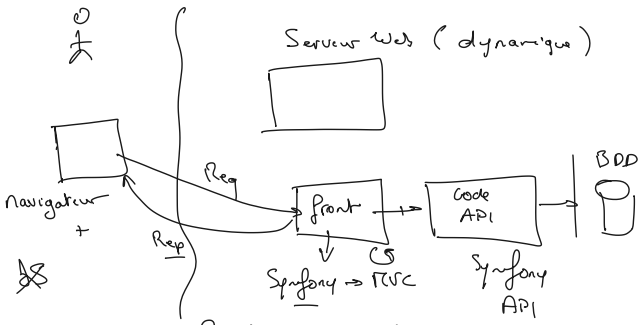
n Process = n Thread = n coeurs

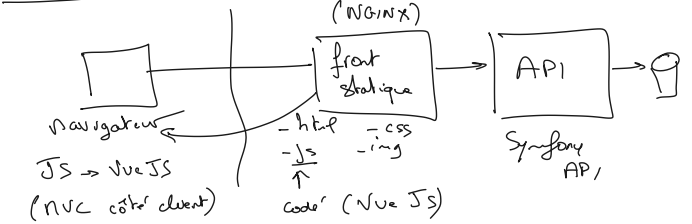
⚠ 1 Process par défaut



Modèle synchrone en série

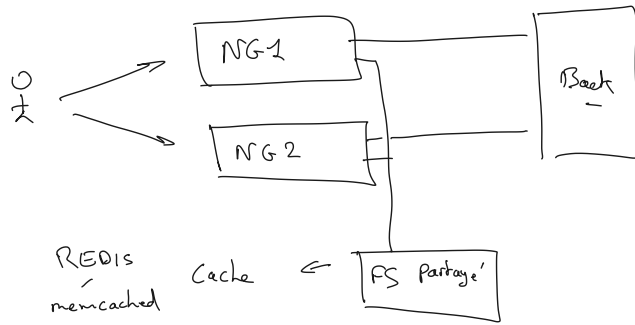






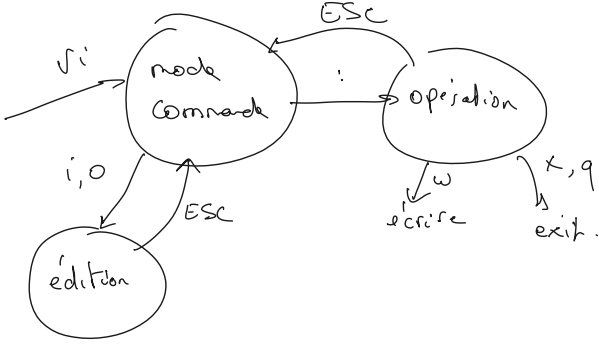
Performance → - Conso CPU pour un calcul donné
- Empreinte mémoire -
- (IO)

1) Code natif → C/C++
2) Code géré (managé) → Java Python .net
3) Script → JS (Php précompilé)
Bash → interprété en fur et à mesure -



- cluster - ensemble de machine qui ont connaissance des autres membres -
- ferme - ensemble de machines sans connaissance des autres membres -

---



sources : /usr/src
en root

auto/confiure -> check des prérequis
ubuntu : apt install build-essentials

auto/configure --prefix=/opt/nginx

auto/configure --help

```
  --help                         print this message

  --prefix=PATH                  set installation prefix
  --sbin-path=PATH               set nginx binary pathname
  --modules-path=PATH              set modules path
  --conf-path=PATH               set nginx.conf pathname
  --error-log-path=PATH            set error log pathname
  --pid-path=PATH                set nginx.pid pathname
  --lock-path=PATH               set nginx.lock pathname

  --user=USER                    set non-privileged user for
                               worker processes
```

Objectif :
auto/configure --prefix=/opt/nginx/1.20 --with-http_ssl_module
...
Configuration summary
 + using system PCRE library
 + using system OpenSSL library
 + using system zlib library

 nginx path prefix: "/opt/nginx/1.20"
 nginx binary file: "/opt/nginx/1.20/sbin/nginx"
 nginx modules path: "/opt/nginx/1.20/modules"
 nginx configuration prefix: "/opt/nginx/1.20/conf"
 nginx configuration file: "/opt/nginx/1.20/conf/nginx.conf"
 nginx pid file: "/opt/nginx/1.20/logs/nginx.pid"
 nginx error log file: "/opt/nginx/1.20/logs/error.log"
 nginx http access log file: "/opt/nginx/1.20/logs/access.log"
 nginx http client request body temporary files: "client_body_temp"
 nginx http proxy temporary files: "proxy_temp"
 nginx http fastcgi temporary files: "fastcgi_temp"
 nginx http uwsgi temporary files: "uwsgi_temp"
 nginx http scgi temporary files: "scgi_temp"

make -j <n>
make install

Juste après compilation :
root@u20:/opt/nginx/1.20# tree .
.
├── conf
│ ├── fastcgi.conf
│ ├── fastcgi.conf.default
│ ├── fastcgi_params
│ ├── fastcgi_params.default
│ ├── koi-utf
│ ├── koi-win
│ ├── mime.types
│ ├── mime.types.default
│ ├── nginx.conf
│ ├── nginx.conf.default
│ ├── scgi_params
│ ├── scgi_params.default
│ ├── uwsgi_params
│ ├── uwsgi_params.default
│ └── win-utf
├── html
│ ├── 50x.html
│ └── index.html
├── logs
└── sbin
└── nginx

SIGKILL : l'OS tue le process (violent)
SIGQUIT : on lui demande de s'arrêter (respectueux)
SIGTERM : QUIT et Kill si pas de réponse.

ps -faux | grep nginx
kill -15 38067

Modèle de fichier de service :
https://www.nginx.com/resources/wiki/start/topics/examples/systemd/

docker container run -d -p 8080:80 --name cyan nboost/webtest Cyan

docker container ls
docker container rm -f cyan

Codes de status HTTP :
https://developer.mozilla.org/fr/docs/Web/HTTP/Status

Utiliser les virtual hosts :

```
server {
      listen 80;
      server_name srv1 srv1.domain.com;    // filtrage par champ hostname de la requête
      location / {
         root /opt/nginx/1.20/srv1;
         index index.html;
      }
   }

   server {
      listen      80 default_server;      ( tout le reste "catch-all" )
      server_name  localhost;
```

Exercice :
"Laisser" le site par défaut pour tout ce qui ne matchera pas
ET Créer deux sites pour :
site1 et site1.domain.com
site2 et site2.domain.com

Contrôler les sites, et les autres adresses (par exemple site3 enverra sur le site par défaut)

Exercice 2 :
Pour un domaine donné, servir des sites web distincts (au moins 2)
qui sont hébergées dans des dossiers différents.

```
server {
      listen 80;
      server_name srv1 srv1.domain.com;
      location /site1 {
         root /opt/nginx/1.20/srv1;
         index index.html;
      }
   }
```

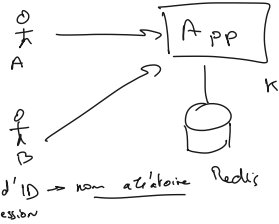
TUTORIELS SERVER WORLD NGINX :
https://www.server-world.info/en/note?os=Ubuntu_16.04&p=nginx&f=9

```
location ~ \.php$ {
            include fastcgi.conf;
            fastcgi_pass unix:/var/run/php/php7.4-fpm.sock;
      }
```

pcre Quick reference :
https://www.pcre.org/current/doc/html/pcre2syntax.html

Sessions

1 App utilise des clef - valeurs (variable = valeur)
Par client (normal)



K-V Session

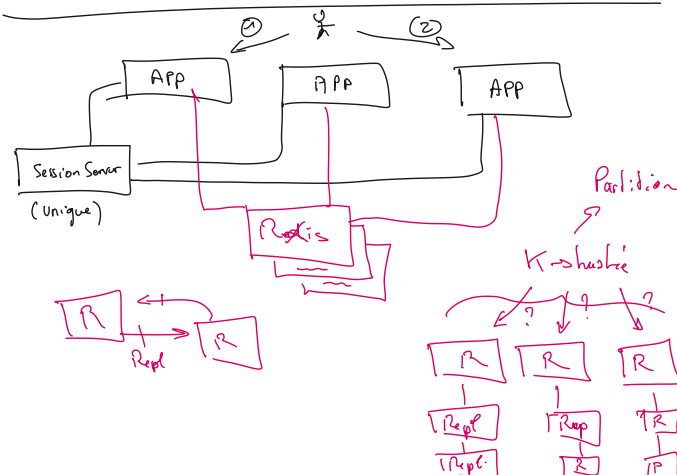
client	clef	Valeur
A	nom	A
A	Panier	2 articles
B	nom	B
B	Panier	3 articles

K-V pour A

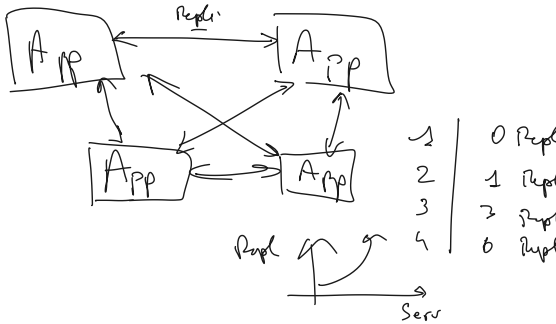
liste des clefs

Données de Session

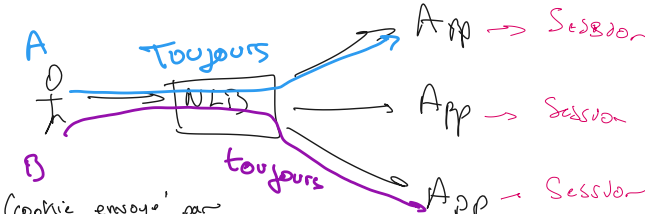
Redis → LE Serveur K-V Par excellence



IBM WebSphere



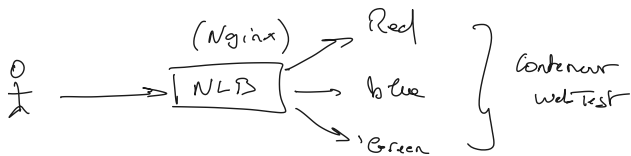
Compromis = Sticky Session



(cookie envoyé par NGinx pour le sticky session)

Exercice

avec 2 navigateurs différents (pour les cookies)
ou via le in Private



→ Tester – Round Robin (par défaut)
– " " avec pondération
– Sticky Session

Tester ce qu'il se passe quand un Conteneur Web Test Tombe
↳ activer le health check pour le sortir

Mise en cache et direct IO on :

```
proxy_cache_path  /var/nginx/cache
    #levels=1:1
    keys_zone=STATIC:1m    inactive=1m  max_size=1g;

    server { # simple load balancing
        listen          80;
        server_name     u20.stage.lan;
        #access_log     logs/big.server.access.log main;

        location / {
            proxy_pass      http://app;
#           proxy_set_header    Host $host;
            proxy_buffering     on;
            proxy_cache         STATIC;
            proxy_cache_valid   any  1s;
            aio on;
            directio      512;
            output_buffers 1 128k;
            proxy_cache_use_stale  error timeout invalid_header updating
                        http_500 http_502 http_503 http_504;
        }
    }
```

K8s

Iaas

VM (lourde) → Paas

Conteneur - - -> Déployés dans des VMs

Php

Php

Php

Php

K8s

-> Manuel

-> Automatique -> Règle de scale In/Out

-> charge

-> Déploie + de Conteneurs

-> Noeuds ?

↳ Déploiement de Noeuds +

AWS

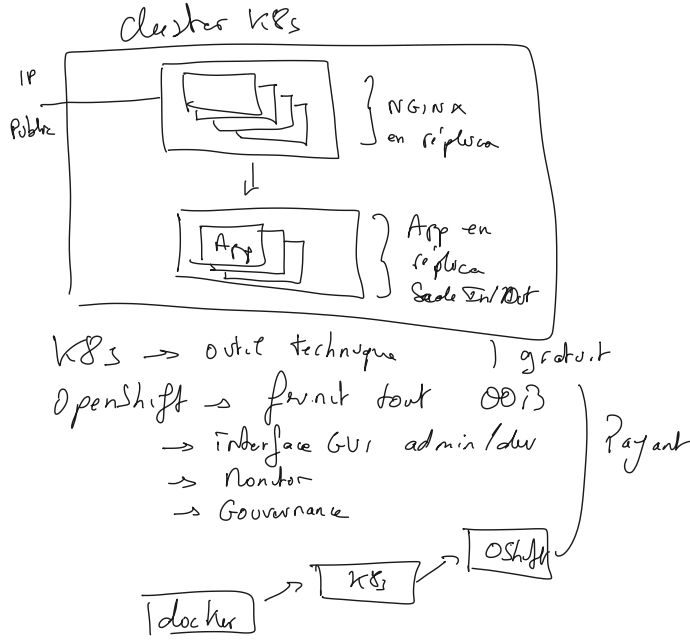
K8s -> gratuit / pas de licence à payer

Paye les VMs

Conteneurise tes Apps

1 IP / Port -> Service sous K8s

(privé) ↘ LoadBalancer géré par K8s



Best practise Symfony avec un NGINX en front
en fast_cgi via php_fpm
https://symfony.com/doc/current/setup/web_server_configuration.html#nginx

TUTOS :
https://techexpert.tips/fr/nginx-fr/nginx-installation-docker/

Installation d'un symfony pour faire des tests :

Installer les prérequis php, symfony dernière version :
(retirer '7.4' pour passer simplement à la dernière version)
apt update && apt -y upgrade
apt install -y php7.4-mbstring php7.4-xmlrpc php7.4-soap php7.4-gd php7.4-xml php7.4-cli php7.4-zip php7.4-mysql php7.4-pgsql php7.4-curl php7.4-sqlite3

Installer composer :
php -r "copy('https://getcomposer.org/installer', 'composer-setup.php');"
php -r "if (hash_file('sha384', 'composer-setup.php') === '906a84df04cea2aa72f40b5f787e49f22d4c2f19492ac310e8cba5b96ac8b64115ac402c8cd292b8a03482574915d1a8') { echo 'Installer verified'; } else { echo 'Installer corrupt'; unlink('composer-setup.php'); } echo PHP_EOL;"
php composer-setup.php
php -r "unlink('composer-setup.php');"
mv composer.phar /usr/local/bin/composer

Installer Symfony :
echo 'deb [trusted=yes] https://repo.symfony.com/apt/ /' | sudo tee /etc/apt/sources.list.d/symfony-cli.list
sudo apt update
sudo apt install symfony-cli

symfony -V doit afficher une version 5 (la version 6 nécessite php 8)
sudo apt install symfony-cli=5.4.1

Créer un site : (ne pas être root !)
mkdir mysite && cd mysite

Créer un projet vide :
#symfony local:new --webapp .
git clone -b v1.8.0 https://github.com/symfony/demo.git (pour récupérer la version 1.8 de la démo, compatible symfony 5 et php7)
cd demo

composer install

Lancer via le serveur de composer :
symfony local:server:start --no-tls --port=8000 --allow-http