

<b>Gestion du référencement</b>	<b>2</b>
1 - Créer un fichier robots.txt	2
2 - Créer un fichier sitemap.xml	2
3 - Configurer un fichier .htaccess	2
4 - Lancer l'indexation de votre site dans la Google Search Console	2
<b>Gérer la mise en production</b>	<b>3</b>
1 - Configurer un script pour la mise en production	3
1 - Se connecter sur le serveur en ssh	3
2 - Installer composer sur votre serveur	3
3 - Installer npm	3
4 - Configurer une clef rsa	3
5 - Créer le dossier dans lequel votre projet sera déposé	4
6 - Créer le script deploy.sh qui exécutera les commandes suivantes:	4
7 - Configurer la base de données	4
<b>Mise en production avec Rsync</b>	<b>5</b>
<b>Aide à la console :</b>	<b>5</b>
<b>Le fichier deploy.sh</b>	<b>6</b>
<b>Configurer son MAILER_DSN avec GMAIL</b>	<b>8</b>
📌 Étape 1 : Activer l'authentification en deux étapes	8
📌 Étape 2 : Générer un mot de passe d'application	8
📌 Étape 3 : Configurer MAILER_DSN dans .env	8
<b>Rappel de la procédure complète pour mettre en production ses dernières modifications :</b>	<b>9</b>
<b>Rappel de la procédure complète pour mettre en production son projet :</b>	<b>9</b>
A ne pas oublier pour éviter les erreurs serveurs :	9

# Gestion du référencement

## 1 - Créer un fichier robots.txt

C'est un fichier de configuration pour le ou les bots Google. C'est un fichier qui permet de donner des instructions aux robots d'indexation Google. C'est pour lui simplifier la tâche et potentiellement lui dire ce qu'il doit faire (indexer) et pas faire (ne pas indexer). Ex: n'indexe pas (ne référence pas) mon backoffice, mon js, mon style...

A déposer à la racine de votre projet.

## 2 - Créer un fichier sitemap.xml

C'est un fichier xml, qui permet de faire un plan de tous les urls de votre site, pour faciliter l'indexation (le référencement) de votre site par Google. Google en a besoin pour comprendre et pouvoir analyser plus facilement votre site. C'est un plan, l'architecture de votre site. On en a besoin si on veut demander à Google de commencer tout de suite le référencement de notre site. Sans ça, le référencement peut prendre des mois.

A déposer à la racine de votre projet.

=> <https://www.xml-sitemaps.com/>

## 3 - Configurer un fichier .htaccess

C'est un fichier de configuration serveur, il permet par exemple de gérer des redirection, de configurer les urls etc... C'est lui qu'on utilise quand on veut par exemple que si on tape <http://www.academiews.fr> ça redirige vers <https://www.academiews.fr>.

A déposer à la racine de votre projet.

=> composer require symfony/apache-pack

## 4 - Lancer l'indexation de votre site dans la Google Search Console

=> <https://search.google.com/search-console/>

# Gérer la mise en production

## 1 - Configurer un script pour la mise en production

Cette partie concerne l'industrialisation du code, c'est à dire toute la configuration liée aux tests, à la mise en production de votre projet.

---

### 1 - Se connecter sur le serveur en ssh

user@server\_ssh  
S'assurer d'être en PHP 8+

ex : ssh [djbpkms@ssh03.cluster43.aaa.hosting.ovh.fr](mailto:djbpkms@ssh03.cluster43.aaa.hosting.ovh.fr)

---

### 2 - Installer composer sur votre serveur

=> <https://getcomposer.org/download/>

---

### 3 - Installer npm

```
curl -fsSL https://raw.githubusercontent.com/nvm-sh/nvm/v0.39.3/install.sh | bash
source ~/.bashrc
command -v nvm
nvm install 18
nvm use 18
node -v
npm -v
```

---

### 4 - Configurer une clef rsa

```
mkdir -p ~/.ssh
chmod 700 ~/.ssh
```

=> ces deux lignes sont à faire si la ligne du dessous fonctionne pas

```
ssh-keyscan -t rsa github.com >> ~/.ssh/known_hosts
```

=> permet d'autoriser GitHub sur votre serveur

```
ssh-keygen -t rsa -b 4096 -C "web.start.contact@gmail.com"
```

=> génère une clef rsa qu'il faudra connecter à Github

```
eval "$(ssh-agent -s)" ssh-add ~/.ssh/id_rsa
```

=> Active l'agent SSH et ajoute ta clé

```
ssh-add -l
```

=> Vérifie que la clé a bien été ajoutée

## MISE EN PRODUCTION

```
cat ~/.ssh/id_rsa.pub
```

=> Affiche ta clef publique

Va sur [GitHub](#) → [SSH Keys](#)

Clique sur «**Settings => deploy keys**»

**Colle la clé** et donne-lui un nom (ex: "Serveur OVH")

**Clique sur "Add SSH key"**

---

### 5 - Créer le dossier dans lequel votre projet sera déposé

---

### 6 - Créer le script deploy.sh qui exécutera les commandes suivantes:

- git pull (git checkout si première fois que tu importes le projet)
- composer update
- npm run build
- php bin/console doctrine:migrations:migrate (fera un database:create si la bdd

n'existe pas)

Pour exécuter le script (A faire une fois) :

chmod +x deploy.sh : donner les droits d'exécution sur notre fichier  
./deploy.sh : exécuter le script

La commande à lancer :

```
ssh djbpkms@ssh03.cluster43.aaa.hosting.ovh.fr "cd /test && ./deploy.sh"
```

---

### 7 - Configurer la base de données

Créer une base de données dans votre hébergement

Configurer le .env sur votre serveur de production

```
DATABASE_URL="mysql://<nomUtilisateur>:<mdp>@<adresse_bdd>:3306/<nom_bdd>?  
serverVersion=8.0"
```

# Mise en production avec Rsync

## 1 - Mise en production avec Rsync

=> [youtube.com/watch?v=qJN9mb8fjDM](https://www.youtube.com/watch?v=qJN9mb8fjDM)

npm run build

rsync -av ./ <user>@<host>:~/<dossier\_du\_projet> --include=public/build --include=public/.htaccess --exclude-from=.gitignore --exclude="\*"

se connecter au serveur

aller dans le dossier du projet

../composer.phar update

database:create

migrate

## Aide à la console :

mkdir <dossier> : créer un dossier

rm <fichier> : supprimer un fichier

touch <nom\_fichier> : créer un fichier

vim (ou nano) <nom\_fichier> : modifier un fichier

esc + : + x : sauvegarder et quitter le fichier

i : modifier le fichier

## Le fichier deploy.sh

```
#!/bin/bash

# Activer le mode strict pour arrêter en cas d'erreur
set -e

# Définir le dossier du projet (où le script est exécuté)
PROJECT_DIR="$(pwd)"
GIT_REPO="git@github.com:webstartMaterial/AFPA_SYMFONY_2025.git"
COMPOSER_PATH="$PROJECT_DIR/../composer.phar"

echo "📁 Répertoire du projet : $PROJECT_DIR"

# Vérifier si le dossier contient déjà un dépôt Git
if [ -d "$PROJECT_DIR/.git" ]; then
    echo "📁 Le projet existe déjà, mise à jour avec Git pull..."
    cd "$PROJECT_DIR"

    # Sauvegarder temporairement les modifications locales
    git stash push -m "Sauvegarde temporaire" --keep-index

    # Mettre à jour le repo sans toucher au fichier .env
    git pull origin main

    # Restaurer les modifications locales
    git stash pop || echo "ℹ️ Aucun changement à restaurer"
else
    echo "🆕 Le dossier existe mais n'est pas un repo Git."

    # Option 1 : Supprimer et re-cloner (⚠️ SUPPRIME TOUT)
    # echo "⚠️ Suppression du dossier existant et clonage du
    projet..."
    # rm -rf "$PROJECT_DIR"
    # git clone "$GIT_REPO" "$PROJECT_DIR"

    # Option 2 : Ajouter Git si nécessaire
    echo "📦 Initialisation Git et récupération du dépôt..."
    cd "$PROJECT_DIR"
    git init
    git remote add origin "$GIT_REPO"
    git fetch origin
    git checkout -t origin/main
fi
```

## MISE EN PRODUCTION

```
# Mise à jour des dépendances PHP
echo "📦 Mise à jour des dépendances PHP avec Composer..."
php "$COMPOSER_PATH" update --no-interaction --optimize-autoloader

# Construire les assets avec NPM
echo "⚙️ Construction des assets avec NPM..."
npm install
npm run build

# Vérifier si la base de données existe
echo "🔍 Vérification de l'existence de la base de données..."
if php bin/console doctrine:database:exists --no-interaction; then
    echo "✅ La base de données existe déjà."
else
    echo "🚀 La base de données n'existe pas, création en cours..."
    php bin/console doctrine:database:create --no-interaction
fi

# Exécuter les migrations
echo "📁 Migration de la base de données..."
php bin/console doctrine:migrations:migrate --no-interaction

echo "✅ Déploiement terminé avec succès !"
```

## Configurer son MAILER\_DSN avec GMAIL

Google bloque souvent les connexions SMTP standard, sauf si tu génères un **mot de passe d'application**.



### Étape 1 : Activer l'authentification en deux étapes

1. Va sur Google Sécurité.
2. Active l'**authentification en deux étapes** si ce n'est pas encore fait.



### Étape 2 : Générer un mot de passe d'application

1. Va sur Gestion des mots de passe d'application. (<https://myaccount.google.com/apppasswords?>)
2. Ajouter un nom comme `Symfony Mailer`.
3. Google va générer un **mot de passe unique** (ex: `abcd efgh ijkl mnop`).
4. Remplacer les espaces dans le mdp par `%20`
5. Copier ce mdp dans votre fichier `.env` en suivant l'étape 3



### Étape 3 : Configurer MAILER\_DSN dans `.env`

Ajoute cette ligne dans ton fichier `.env` Symfony :

```
MAILER_DSN=smtp://your-email@gmail.com:your-app-password@smtp.gmail.com:587?  
encryption=tls&auth_mode=login
```



Remplace :

- `your-email@gmail.com` → par ton adresse Gmail.
- `your-app-password` → par le **mot de passe d'application** généré.
- `smtp.gmail.com` → hôte SMTP de Gmail.
- `587` → port SMTP avec chiffrement TLS.



## **Rappel de la procédure complète pour mettre en production ses dernières modifications :**

```
Git add -A  
Git commit -m 'modif'  
Git push
```

```
Se connecter au serveur en ssh  
Aller dans le répertoire ou se trouve votre dossier  
./deploy.sh
```

## **Rappel de la procédure complète pour mettre en production son projet :**

```
Git add -A  
Git commit -m 'modif'  
Git push  
Se connecter au serveur en ssh  
Créer le répertoire dans lequel tu vas mettre ton projet  
Touch deploy.sh  
Vim deploy.sh  
Chmod +x deploy.sh  
  
./deploy.sh
```

## **A ne pas oublier pour éviter les erreurs serveurs :**

.gitignore (en local) ajouter /.env et /deploy.sh car ces fichiers seront en conflits sur votre serveur car parfois nous allons les créer/modifier