

BTS SIO OPTION SLAM

DOCUMENTATION TECHNIQUE



ATELIER DE PROFESSIONNALISATION 3

CCI APPRO

DOCUMENTATION TECHNIQUE

[Résumé](#)

Guide d'installation et de configuration de l'application.

SOMMAIRE

1. Prérequis d'installation	3
1.1. Logiciels.....	3
1.2. Extensions PHP	3
1.1.1. Pour une mise en production	3
1.1.2. Pour le développement	3
2. Procédure d'installation et de mise à jour	4
2.1. Installation de l'application	4
2.1.1. Mise à jour des librairies.....	4
2.1.2. Installation d'Apache - PHP - MariaDB – xdebug.....	4
2.1.3. Configuration du Virtual Host vers /var/www/CCI-Appro	5
2.1.4. Configuration de MariaDB avec BDD et user	8
2.1.5. Installation de NodeJs – NPM	8
2.1.6. Installation de Composer	9
2.1.7. Installation du projet en Laravel 9	10
2.2. Mise à jour de l'application.....	13
3. Normes de codage	14
4. Migration des données	16
4.2. Guide d'utilisation.....	16
5. Conception de la base de données	17
5.1. Dictionnaire des données	17
5.2. Modèle conceptuel des données.....	17
5.3. Modèle logique des données	18
5.4. Diagramme de classes	18
5.5. Diagramme de contexte statique.....	19
5.6. Diagramme de cas d'utilisation	19
5.7. Description textuelle complète du cas d'utilisation	20
5.7.1. 1er cas d'utilisation : Sommaire d'identification des commandes des fournitures	20
Description des scénarios	20
5.7.2. 2eme cas d'utilisation : Gestion de l'état des commandes	22
Description des scénarios	22
5.7.3. 3eme cas d'utilisation : Exports Excel.....	25
Description des scénarios	25
6. Tests unitaires	26

6.1. Lancement des tests :	26
6.2. Analyse du code :	26
7. Configuration de l'envoi d'emails	27
7.1. Postfix	27
7.1.1. Installation	27
7.1.2. Configuration	27
7.1.3. Eléments complémentaires	30
7.2. Supervisor	30
7.2.1. Installation	31
7.2.2. Configuration	31
7.2.3. Eléments complémentaires	32

1. Prérequis d'installation

- Dans le cas où les assets de l'application compilés sont présents sur le dépôt du projet et que la mise à jour du projet se fait via git, la machine de production ne nécessite pas l'installation de NodeJs et npm. Dans le cas contraire, ils seront nécessaires au bon fonctionnement de l'application.

1.1. Logiciels

- **MariaDB** : Ver 15.1 Distrib 10.3.34-MariaDB
- **PHP** : 8.1.12
- **npm** : 9.1.2
- **NodeJs** : 18.12.0

1.2. Extensions PHP

1.1.1. Pour une mise en production

- BCMath
- CType
- Fileinfo
- JSON
- Mbstring
- OpenSSL
- PDO
- Tokenizer
- XML

1.1.2. Pour le développement

- xdebug

2. Procédure d'installation et de mise à jour

2.1. Installation de l'application

Lors du déploiement de la machine virtuelle qui va héberger le site pour le développement, vous pouvez effectuer dans l'ordre les commandes suivantes :

2.1.1. Mise à jour des librairies

```
sudo apt update && sudo apt upgrade -y
```



```
# Mise à jour des librairies  
sudo apt update && sudo apt upgrade -y
```

2.1.2. Installation d'Apache - PHP - MariaDB – xdebug

```
sudo apt install software-properties-common -y
```

```
sudo add-apt-repository ppa:ondrej/php -y
```

```
sudo apt update -y && sudo apt upgrade -y
```

2.1.2.1. Installation d'Apache et MariaDB

```
sudo apt install apache2 -y mariadb-server -y mariadb-client -y
```

2.1.2.2. Installation de PHP 8.1 et ses paquets

```
sudo apt install php8.1-
```

```
{common,mysql,xml,xmldrpc,xdebug,curl,gd,imagick,cli,dev,imap,m  
bstring,opcache,soap,zip,redis,intl,bcmath,fpm,ldap,bz2,pgsql,  
cgi} -y
```

2.1.2.3. Activation du proxy_fcgi et php8.1-fpm

```
sudo a2enmod proxy_fcgi setenvif
```

```
sudo a2enconf php8.2-fpm
```

2.1.2.4. Restart Apache

```
sudo systemctl restart apache2
```

A terminal window with a black background and white text. The text shows a series of commands for installing and configuring Apache, PHP 8.1, and MariaDB. The commands are: # Installation d'Apache - PHP - MariaDB - xdebug"; sudo apt install software-properties-common -y; sudo add-apt-repository ppa:ondrej/php -y; sudo apt update -y && sudo apt upgrade -y; # Installation de PHP 8.1 et ses paquets; sudo apt install php8.1 -y php8.1-{common,mysql,xml,xmllrpc,xdebug,curl,gd,imagick,cli,dev,imap,mbstring,opcache,soap,zip,redis,intl,bcmath,fpm,ldap,bz2,pgsql,cgi} -y; # Installation d'Apache et MariaDB; sudo apt install apache2 -y mariadb-server -y mariadb-client -y; # Activation du proxy_fcgi et php8.1-fpm; sudo a2enmod proxy_fcgi setenvif; sudo a2enconf php8.1-fpm; # Restart d'Apache; sudo systemctl restart apache2; The terminal window has three colored dots (red, yellow, green) in the top left corner.

```
# Installation d'Apache - PHP - MariaDB - xdebug";
sudo apt install software-properties-common -y
sudo add-apt-repository ppa:ondrej/php -y
sudo apt update -y && sudo apt upgrade -y
# Installation de PHP 8.1 et ses paquets
sudo apt install php8.1 -y php8.1-
{common,mysql,xml,xmllrpc,xdebug,curl,gd,imagick,cli,
dev,imap,mbstring,opcache,soap,zip,redis,intl,bcmath
,fpm,ldap,bz2,pgsql,cgi} -y;
# Installation d'Apache et MariaDB
sudo apt install apache2 -y mariadb-server -y
mariadb-client -y
# Activation du proxy_fcgi et php8.1-fpm
sudo a2enmod proxy_fcgi setenvif;
sudo a2enconf php8.1-fpm;
# Restart d'Apache
sudo systemctl restart apache2;
```

2.1.3. Configuration du Virtual Host vers /var/www/CCI-Appro

Nous nous rendons dans le répertoire pour les fichiers de configuration d'Apache :

```
cd /etc/apache2/sites-available
```

Nous créons un fichier de configuration appelé `projetv1.conf`

```
sudo nano projetv1.conf
```

Ce fichier va contenir la configuration suivante :

```
<VirtualHost *:80>
```

```
    ServerName dev.ccc-appro.com
```

```
    ServerAdmin webmaster@localhost
```

```
    DocumentRoot /var/www/CCI-Appro/public/
```

```
    ErrorLog \${APACHE_LOG_DIR}/error.log
```

```
    CustomLog \${APACHE_LOG_DIR}/access.log combined
```

```
    <Directory /var/www/CCI-Appro/>
```

```
        Options Indexes FollowSymLinks MultiViews
```

```
        AllowOverride All
```

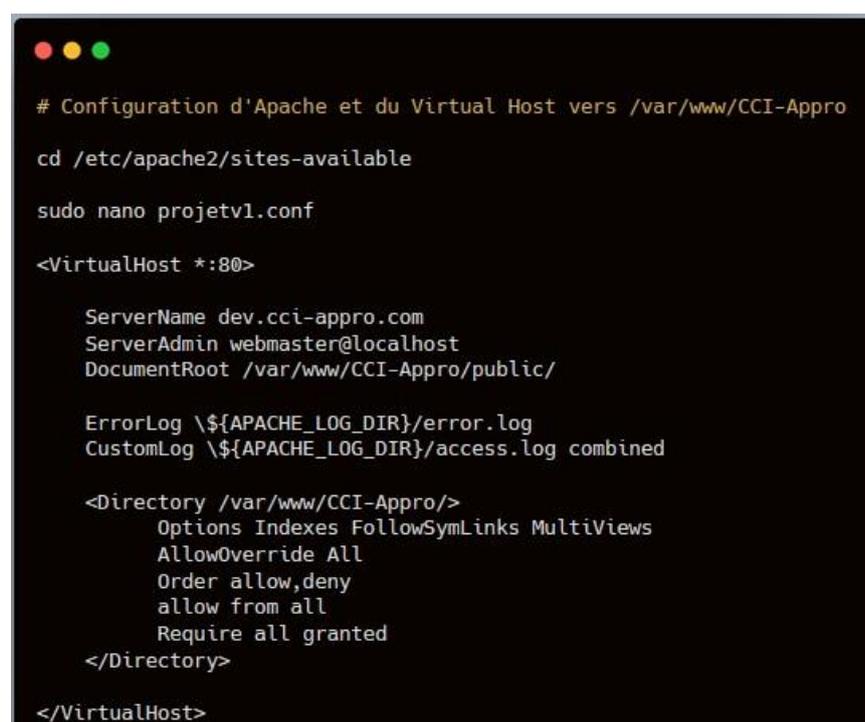
```
        Order allow,deny
```

```
        allow from all
```

```
        Require all granted
```

```
    </Directory>
```

```
</VirtualHost>
```



```
# Configuration d'Apache et du Virtual Host vers /var/www/CCI-Appro
cd /etc/apache2/sites-available
sudo nano projetv1.conf
<VirtualHost *:80>
    ServerName dev.cci-appro.com
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/CCI-Appro/public/
    ErrorLog \${APACHE_LOG_DIR}/error.log
    CustomLog \${APACHE_LOG_DIR}/access.log combined
    <Directory /var/www/CCI-Appro/>
        Options Indexes FollowSymLinks MultiViews
        AllowOverride All
        Order allow,deny
        allow from all
        Require all granted
    </Directory>
</VirtualHost>
```

2.1.3.1. Activation du projetv1.conf

```
sudo a2ensite projetv1.conf
```

2.1.3.2. Désactivation du 000-default.conf

```
sudo a2dissite 000-default.conf
```

2.1.3.3. Activation du module mod_rewrite d'Apache

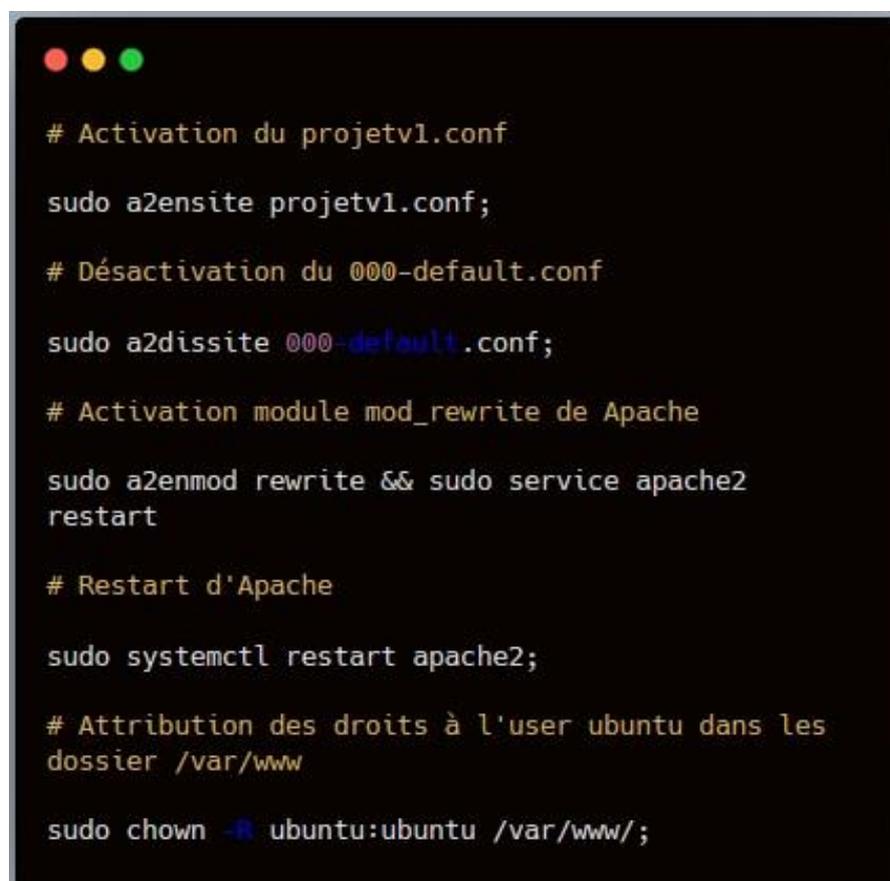
```
sudo a2enmod rewrite && sudo service apache2 restart
```

2.1.3.4. Restart d'Apache

```
sudo systemctl restart apache2
```

2.1.3.5. Attribution des droits à l'utilisateur ubuntu dans le dossier /var/www/

```
sudo chown -R ubuntu:ubuntu /var/www/
```



```
# Activation du projetv1.conf
sudo a2ensite projetv1.conf;

# Désactivation du 000-default.conf
sudo a2dissite 000-default.conf;

# Activation module mod_rewrite de Apache
sudo a2enmod rewrite && sudo service apache2
restart

# Restart d'Apache
sudo systemctl restart apache2;

# Attribution des droits à l'utilisateur ubuntu dans les
dossier /var/www
sudo chown -R ubuntu:ubuntu /var/www/;
```

2.1.4. Configuration de MariaDB avec BDD et user

```
sudo mariadb
```

```
DROP USER IF EXISTS '[pseudo]'@'localhost';
```

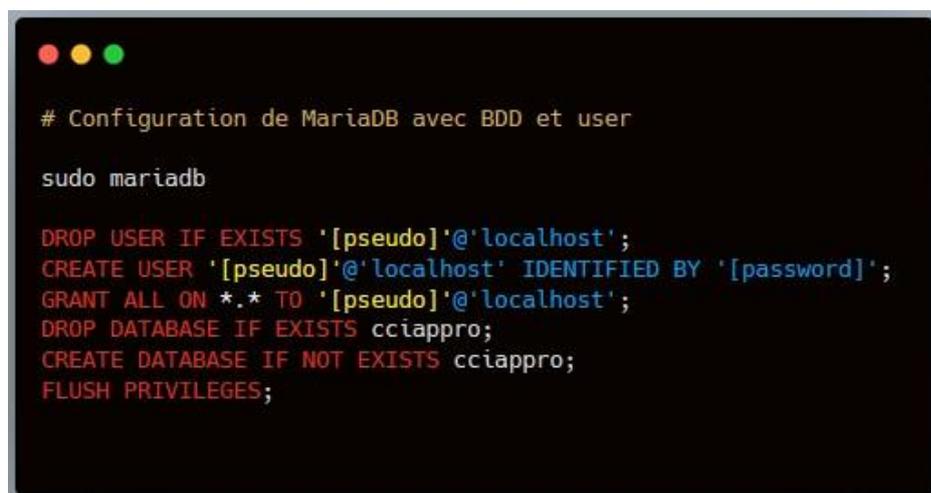
```
CREATE USER '[pseudo]'@'localhost' IDENTIFIED BY '[password]';
```

```
GRANT ALL ON *.* TO '[pseudo]'@'localhost';
```

```
DROP DATABASE IF EXISTS cciappro;
```

```
CREATE DATABASE IF NOT EXISTS cciappro;
```

```
FLUSH PRIVILEGES;
```

A screenshot of a terminal window with a black background and white text. The window title is "# Configuration de MariaDB avec BDD et user". The terminal shows the following commands: "sudo mariadb", "DROP USER IF EXISTS '[pseudo]'@'localhost';", "CREATE USER '[pseudo]'@'localhost' IDENTIFIED BY '[password]';", "GRANT ALL ON *.* TO '[pseudo]'@'localhost';", "DROP DATABASE IF EXISTS cciappro;", "CREATE DATABASE IF NOT EXISTS cciappro;", and "FLUSH PRIVILEGES;". The terminal output is not visible, only the commands are shown.

```
# Configuration de MariaDB avec BDD et user  
sudo mariadb  
DROP USER IF EXISTS '[pseudo]'@'localhost';  
CREATE USER '[pseudo]'@'localhost' IDENTIFIED BY '[password]';  
GRANT ALL ON *.* TO '[pseudo]'@'localhost';  
DROP DATABASE IF EXISTS cciappro;  
CREATE DATABASE IF NOT EXISTS cciappro;  
FLUSH PRIVILEGES;
```

2.1.5. Installation de NodeJs – NPM

2.1.5.1. Update NodeJs lastest version v18.9.0

```
curl -fsSL https://deb.nodesource.com/setup_18.x | sudo -E  
bash -
```

```
sudo apt-get install nodejs -y
```

2.1.5.2. Compiler des modules complémentaires natifs à partir de npm

```
sudo apt install build-essential -y
```

2.1.5.3. Update NPM latest

```
sudo npm install -g npm@latest
```



```
# Installation de Nodejs - NPM
# Update NodeJS latest version v17.9.0
curl -fsSL https://deb.nodesource.com/setup_18.x | sudo -E bash -;
sudo apt-get install nodejs -y;
# Compiler des modules complémentaires natifs à partir de npm
sudo apt install build-essential -y;
# Update NPM latest
sudo npm install -g npm@latest;
```

2.1.6. Installation de Composer

- cd ~
- sudo apt install php-cli unzip -y
- curl -sS https://getcomposer.org/installer -o composer-setup.php
- sudo php composer-setup.php --install-dir=/usr/local/bin --filename=composer
- php -r "copy('https://getcomposer.org/installer', 'composer-setup.php');"
- HASH="\$(wget -q -O - https://composer.github.io/installer.sig)"
- php -r "if (hash_file('SHA384', '/tmp/composer-setup.php') === '\$HASH') { echo 'Installer verified'; } else { echo 'Installer corrupt'; unlink('composer-setup.php'); } echo PHP_EOL;"
- php composer-setup.php
- php -r "unlink('composer-setup.php');"

2.1.6.1. Donne les droits à l'utilisateur ubuntu pour l'utilisation de composer

```
sudo chown ubuntu:ubuntu /usr/local/bin/composer
```

```
# Installation de Composer

cd ~;
sudo apt install php-cli unzip -y;

curl -sS https://getcomposer.org/installer -o composer-setup.php;

sudo php composer-setup.php --install-dir=/usr/local/bin
--filename=composer;

php -r "copy('https://getcomposer.org/installer', 'composer-
setup.php');"

HASH="$(wget -q -O - https://composer.github.io/installer.sig)";

php -r "if (hash_file('SHA384', '/tmp/composer-setup.php') ==
'$HASH') { echo 'Installer verified'; } else { echo 'Installer
corrupt'; unlink('composer-setup.php'); } echo PHP_EOL;";

php composer-setup.php;

php -r "unlink('composer-setup.php');"

# Donne les droits à ubuntu pour l'utiliser composer

sudo chown ubuntu:ubuntu /usr/local/bin/composer
```

2.1.7. Installation du projet en Laravel 9

2.1.7.1. Cloner le projet

```
cd /var/www
```

```
git clone git@github.com:CCI-Campus/CCI-Appro.git CCI-Appro
```

Se rendre dans le dossier du projet

```
cd CCI-Appro
```

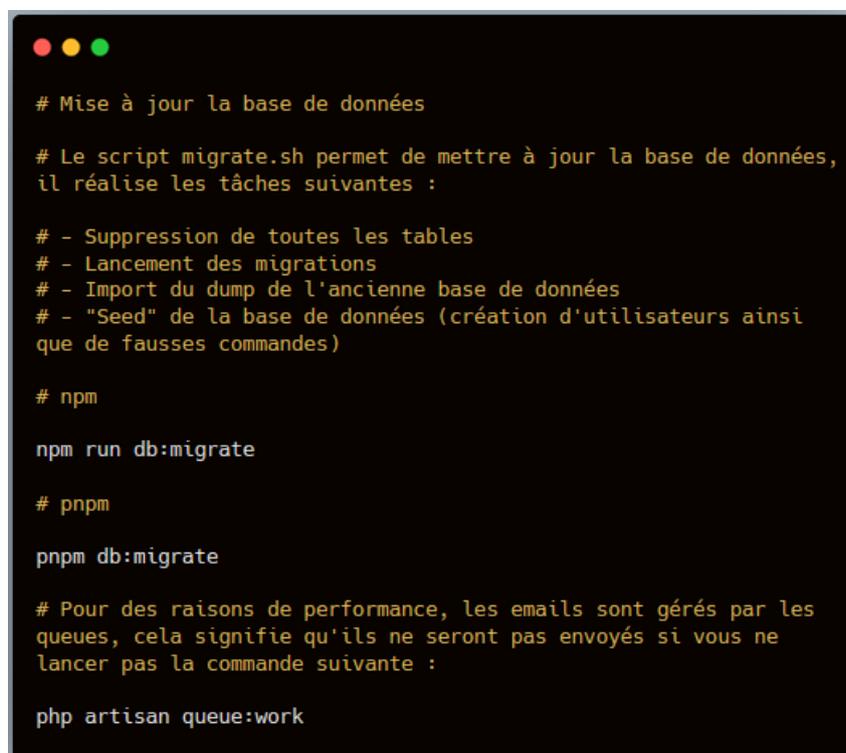
2.1.7.2. Lancer le script de mise en place du projet

Ce script vous demandera différentes informations afin de configurer le `.env` de l'application avec ces données.

- Le type d'environnement (développement/production)
- Le nom de la base de données
- Les identifiants de l'utilisateur MariaDB
- La clé d'API* permettant d'accéder aux identifiants du serveur smtp de gmail (pour un environnement de développement)

* La clé d'API n'est pas disponible sur le github

```
chmod +x setup.sh && bash setup.sh
```



```
# Mise à jour la base de données

# Le script migrate.sh permet de mettre à jour la base de données,
il réalise les tâches suivantes :

# - Suppression de toutes les tables
# - Lancement des migrations
# - Import du dump de l'ancienne base de données
# - "Seed" de la base de données (création d'utilisateurs ainsi
que de fausses commandes)

# npm

npm run db:migrate

# pnpm

pnpm db:migrate

# Pour des raisons de performance, les emails sont gérés par les
queues, cela signifie qu'ils ne seront pas envoyés si vous ne
lancer pas la commande suivante :

php artisan queue:work
```

2.1.7.3. Mise à jour la base de données

Le script **migrate.sh** permet de mettre à jour la base de données, il réalise les tâches suivantes :

- Suppression de toutes les tables
- Lancement des migrations
- Import du dump de l'ancienne base de données
- "Seed" de la base de données (création d'utilisateurs ainsi que de fausses commandes)

```
# npm
```

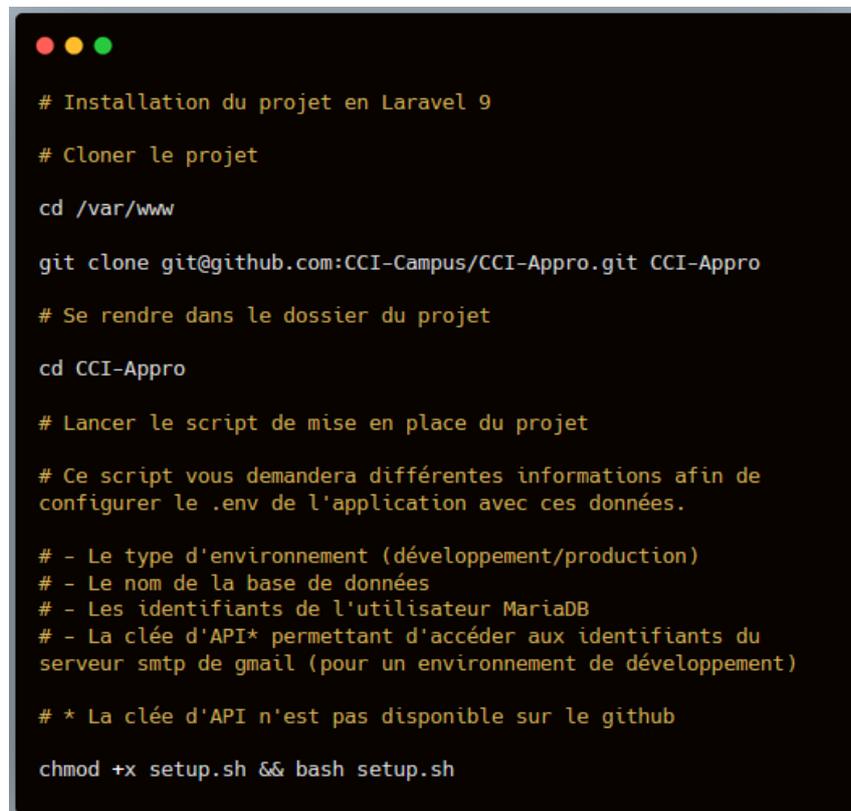
```
npm run db:migrate
```

```
# pnpm
```

```
pnpm db:migrate
```

Pour des raisons de performance, les emails sont gérés par les queues, cela signifie qu'ils ne seront pas envoyés si vous ne lancez pas la commande suivante :

```
php artisan queue:work
```



```
# Installation du projet en Laravel 9
# Cloner le projet
cd /var/www
git clone git@github.com:CCI-Campus/CCI-Appro.git CCI-Appro
# Se rendre dans le dossier du projet
cd CCI-Appro
# Lancer le script de mise en place du projet
# Ce script vous demandera différentes informations afin de
# configurer le .env de l'application avec ces données.
# - Le type d'environnement (développement/production)
# - Le nom de la base de données
# - Les identifiants de l'utilisateur MariaDB
# - La clé d'API* permettant d'accéder aux identifiants du
# serveur smtp de gmail (pour un environnement de développement)
# * La clé d'API n'est pas disponible sur le github
chmod +x setup.sh && bash setup.sh
```

2.2. Mise à jour de l'application

- La mise à jour de l'application s'effectuera par un programme développé lors de notre atelier ([GitHub](#)).

Ce programme permettra la gestion du serveur ainsi que de l'application à l'aide d'une interface en ligne de commande.

- L'administrateur aura alors accès aux fonctionnalités suivantes :
- Informations de la machine
- Gérer les services (voir les statuts, redémarrer un service, ...)
- Voir les versions des logiciels installés
- Redémarrer le serveur
- Accéder au terminal
- Faire un export de la base de données de l'application
- Mettre à jour l'application*¹
- Activer/Désactiver l'état de maintenance de l'application

*¹ La mise à jour du site s'effectuera, pour des raisons de praticité via git.

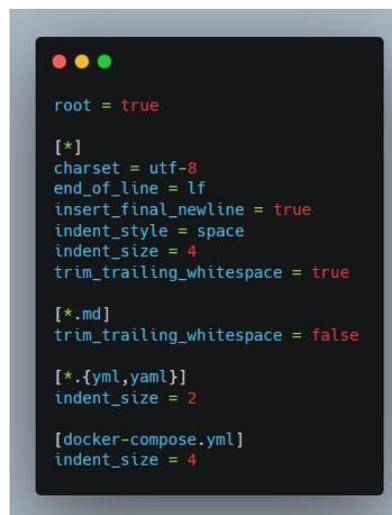
3. Normes de codage

D'après la norme [PSR-12](#), il est nécessaire d'utiliser à la place de la tabulation, quatre espaces pour gérer l'alignement et la mise en page du code.

Cela est géré à l'aide de fichiers tels que

« .editorconfig »,

- Ce fichier instaure les normes de codage du projet pour l'application aux éditeurs.



```
root = true

[*]
charset = utf-8
end_of_line = lf
insert_final_newline = true
indent_style = space
indent_size = 4
trim_trailing_whitespace = true

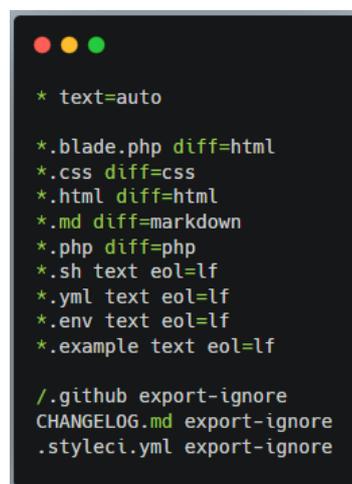
[*.md]
trim_trailing_whitespace = false

[*.{yml,yaml}]
indent_size = 2

[docker-compose.yml]
indent_size = 4
```

Ainsi que le fichier « .gitattributes »,

- Il permet d'effectuer certaines configurations assurant le respect des normes de codage en indiquant ce qui doit être appliqué sur GitHub par exemple.



```
* text=auto

*.blade.php diff=html
*.css diff=css
*.html diff=html
*.md diff=markdown
*.php diff=php
*.sh text eol=lf
*.yml text eol=lf
*.env text eol=lf
*.example text eol=lf

/.github export-ignore
CHANGELOG.md export-ignore
.styleci.yml export-ignore
```

- Il est également nécessaire d'écrire tous les mots clés (tel que : break, const, function...) en minuscule et cela peu importe la version utilisée de php.

Il est important de bien typer les éléments du code à l'aide de la [PHPDoc](#), parexemple :

```
/**
 * @return string|null The file path or null when non-existent.
 */
function getPath()
{
    ...
    return $path;
}
```

Cela permet d'éviter de limiter les possibilités d'erreur à ce niveau.

Sachant que le projet est réalisé avec Laravel, certaines fonctions n'ont pas la possibilité d'être typée à notre niveau.

4. Migration des données

Pour cette version du projet nous avons voulu modifier la structure de la base de données, car celle de la première version ne nous convenait pas.

Cependant le cahier des charges indique que les données doivent, pour la plupart, être réintégrées dans la seconde version du projet. Pour réaliser la migration des données, il faut donc prendre en compte les changements dans la nouvelle base de données.

4.2. Guide d'utilisation

Vous pouvez exécuter ce script depuis un terminal ou depuis l'interface phpMyAdmin.

Les étapes ci-dessous sont à réaliser pour assurer le fonctionnement de la migration :

Cet exemple a été réalisé sur Ubuntu 20.04.

- Lancement des migrations
 - o « php artisan migrate:fresh »
- Entrer dans MariaDB
 - o « sudo mariadb »
- Entrer dans la base de données
 - o « use < db name >
- Importer le fichier de l'ancienne base de données
 - o « source < path to dump file > »
- Importer les fonctions utiles pour l'exécution du script
 - o « source < path to function file > »
- Exécuter le script de migration des données
 - o « source < path to migration script file > »

Le script « migration.sh » présent dans « docs\database\migration\ » permet d'automatiser cela ainsi que de faire un dump de la structure modifiée dans « database\dumps ».

5. Conception de la base de données

Réalisations avec [Looping](#) :

- Diagramme de classes
- Modèle conceptuel des données
- Modèle logique des donnés

Réalisations avec [Drawio](#) :

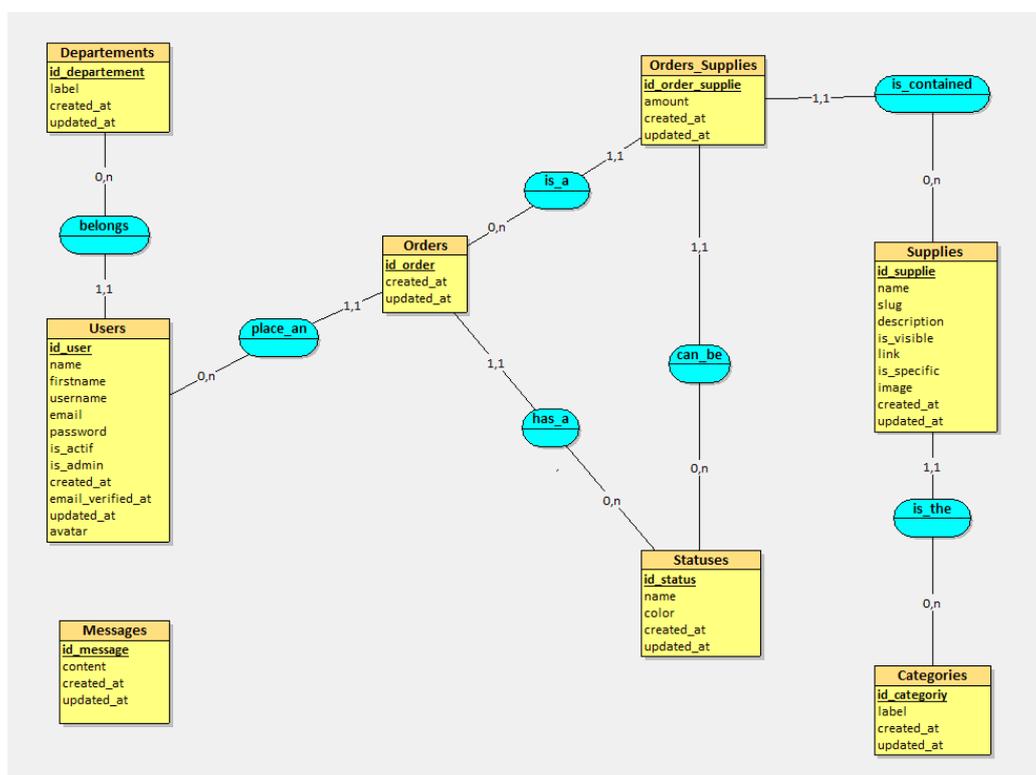
- Diagramme de contexte statique
- Diagramme de cas d'utilisation

5.1. Dictionnaire des données

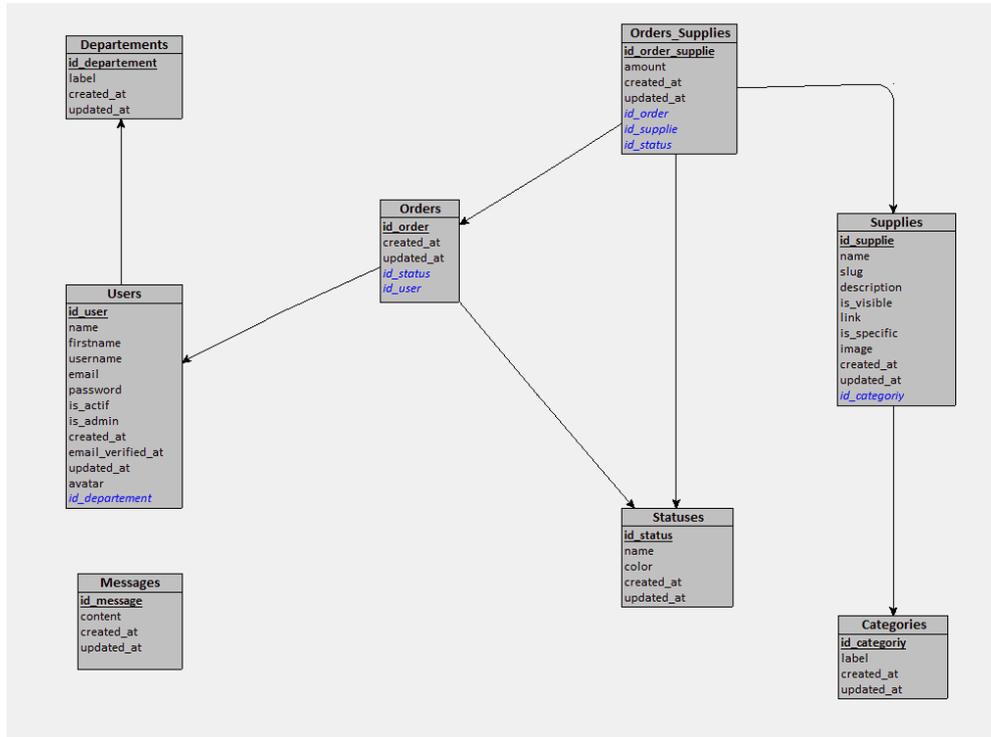
[Dictionnaire des données accessible en cliquant ici.](#)

Le dictionnaire des données est en format PDF.

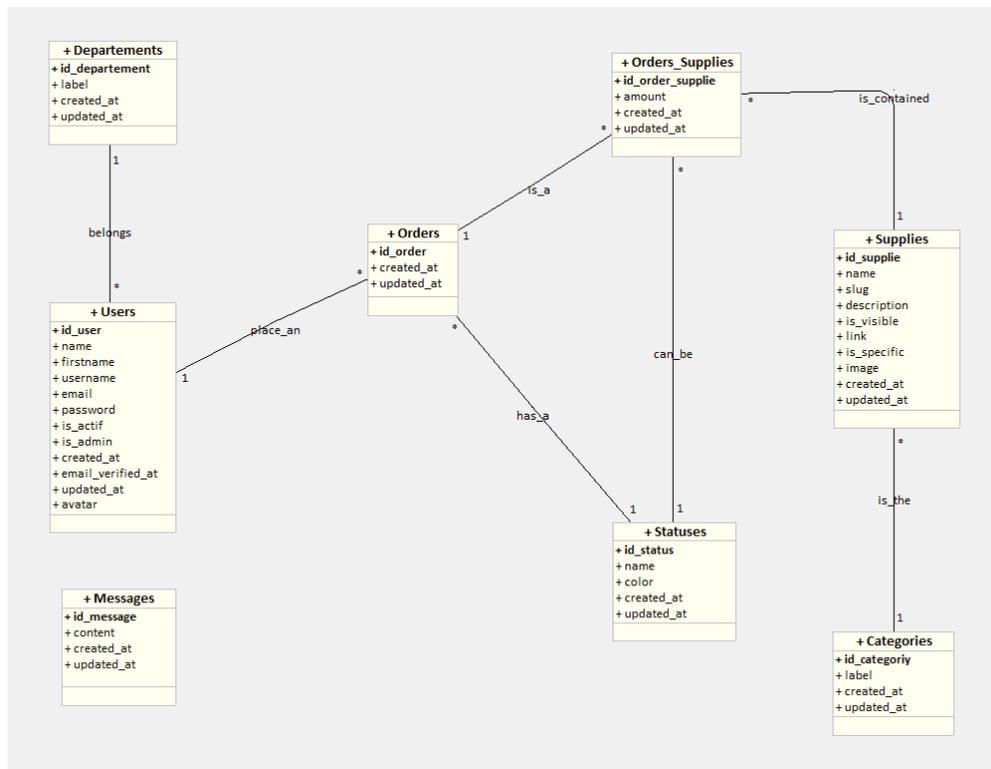
5.2. Modèle conceptuel des données



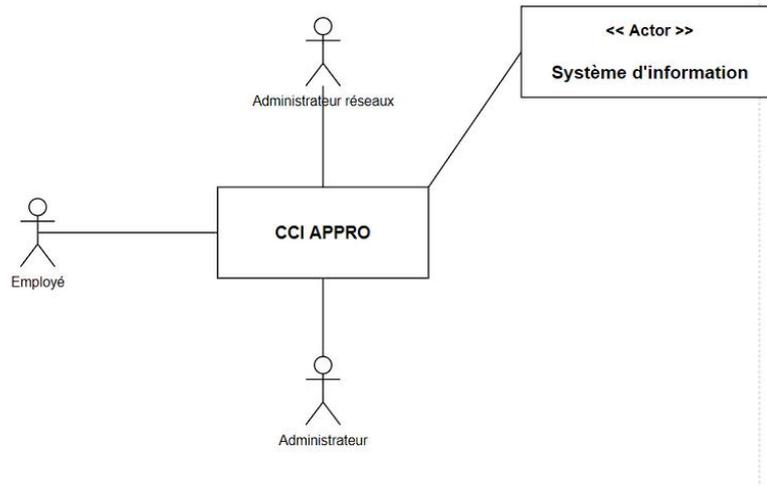
5.3. Modèle logique des données



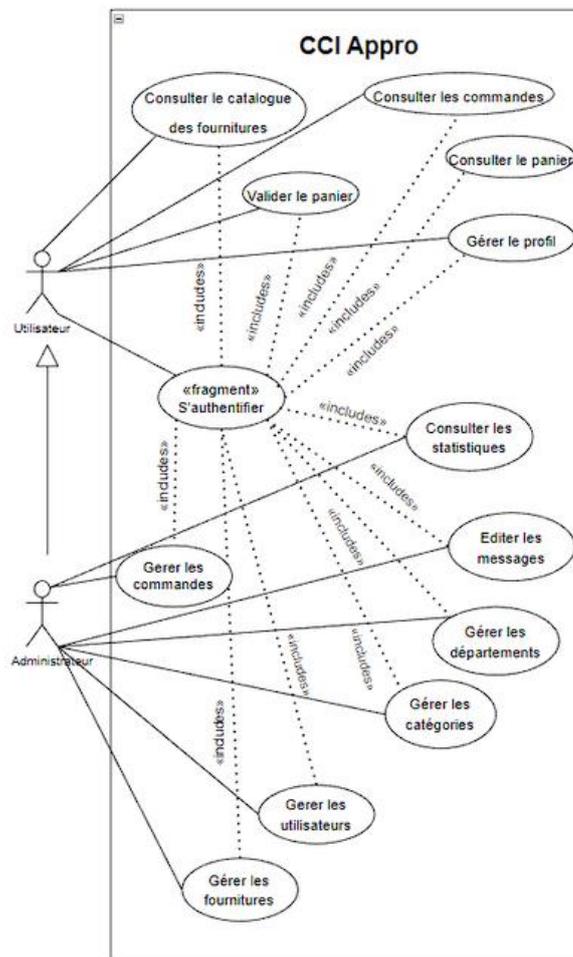
5.4. Diagramme de classes



5.5. Diagramme de contexte statique



5.6. Diagramme de cas d'utilisation



5.7. Description textuelle complète du cas d'utilisation

5.7.1. 1er cas d'utilisation : Sommaire d'identification des commandes des fournitures

Titre : Choix de la fourniture et commande

Résumé : L'utilisateur a la possibilité de se connecter à son Espace Client et consulter le catalogue des fournitures en cliquant le lien dans la nav-bar. Il peut choisir le produit qu'il souhaite, sélectionner la quantité à commander puis ajouter ce dernier au panier. Le panier est consultable depuis le lien dans la nav-bar. L'utilisateur se rend dans le panier et peut valider la commande. La commande sera validée ou non par l'administrateur.

Acteur : Utilisateur

Création : 27/09/22

Date de mise à jour : 02/01/23

Responsable : Vladimir SACCHETTO

Description des scénarios

Préconditions

- L'utilisateur possède une connexion internet stable.
- L'utilisateur est connecté à la plateforme.

Scénario nominal

1. L'utilisateur se rend dans le catalogue des fournitures
2. L'utilisateur consulte la liste des produits disponibles ainsi que le descriptif de ces derniers
3. L'utilisateur sélectionne les produits qu'il souhaite commander ainsi que leur quantité
4. L'utilisateur est notifié par un flash message indiquant que le produit a bien été intégré à son panier
5. L'utilisateur se rend dans son panier
6. L'utilisateur consulte la liste des produits qu'il a sélectionnés et la modifie si besoin
7. L'utilisateur valide sa commande
8. A la validation l'utilisateur est notifié par un mail
9. L'utilisateur suit l'avancement de sa commande sur son espace

10. L'utilisateur reçoit un email lorsque le statut de sa commande change
11. L'utilisateur reçoit un email lorsque le statut d'un produit de sa commande change
12. L'utilisateur reçoit un email lorsque la commande est livrée

Enchaînement alternatif

A1 : L'utilisateur met le produit dans le panier à 0

4. L'utilisateur est notifié par un flash message indiquant que le produit a été retiré de son panier

Le scénario nominal reprend au point 3.

A2 : L'utilisateur ajoute un même produit au panier où le produit existe déjà, le produit dans le panier s'incrémente

A3 : La commande a été refusée par l'administrateur

8. L'administrateur refuse la commande de l'utilisateur.
8. L'utilisateur est notifié par mail que sa commande a été refusée par l'administrateur. Le scénario nominal reprend au point 1.

5.7.2. 2eme cas d'utilisation : Gestion de l'état des commandes

Titre : Gestion des commandes

Résumé : L'administrateur a la possibilité de gérer les commandes créées par des utilisateurs. Il peut changer le statut des commandes ou bien tout simplement les refuser.

Acteur : Administrateur

Création : 03/01/2023

Date de mise à jour : 03/01/2023

Responsable : Alexis HENRY

Description des scénarios

Préconditions

- L'utilisateur possède une connexion internet stable.
- L'utilisateur est connecté à la plateforme.
- L'utilisateur possède les droits d'administration.
- Des commandes ont été réalisées.

Scénario nominal

1. L'administrateur se rend dans la partie administrateur, gestion des commandes.
2. L'administrateur peut filtrer les commandes grâce au filtre présent en haut de la page : il permet le filtre par département, statut ou même recherche sur l'utilisateur ayant effectué la commande.
3. L'administrateur clique sur le bouton qui permet de gérer la commande (icone crayon).
4. L'administrateur arrive sur la page de gestion de la commande voulue.
5. L'administrateur peut voir différentes informations utiles à la gestion de la commande en haut à gauche de la section : utilisateur ayant créé la commande, la date de création ainsi que la date de la dernière modification.
6. Depuis cette page l'administrateur peut exporter la commande sous format Excel à l'aide du lien « Exporter la commande ».
7. En dessous du titre « Fournitures commandées », l'administrateur peut apercevoir les différents produits présents dans la commande. Les produits sont présentés sous la forme : image, nom du produit, quantité, statuts.

8. L'administrateur modifie alors le statut d'un produit en cliquant sur les boutons, il a alors le choix entre les différents boutons présents à hauteur de ladite fourniture : « En attente de validation », « En cours de préparation », « Livrée », « Rejetée ».
9. L'administrateur peut directement annuler la commande entièrement à l'aide du bouton « Ne pas traiter cette commande », situé au bas de la page.

Enchaînement alternatif

A1. L'administrateur modifie le statut d'un produit d'une commande ayant plusieurs produits

A1.1. L'administrateur met toutes les fournitures sur le statut « Rejetée »

9. L'administrateur peut directement annuler la commande entièrement à l'aide du bouton « Ne pas traiter cette commande », situé au bas de la page.

- La commande est alors considérée comme « Rejetée »

A.1.2. L'administrateur met toutes les fournitures sur le statut « Livrée ».

- La commande est alors considérée comme « Livrée ».

A.1.3. L'administrateur indique une fourniture comme « Livrée » et ne modifie pas les autres.

- La commande est considérée comme « En cours de préparation ».

A.1.4. L'administrateur indique que toutes les fournitures sont en « En cours de préparation ».

- La commande est considérée comme « En cours de préparation ».

A.1.5. L'administrateur met deux fournitures sur le statut rejeté ainsi que deux fournitures sur le statut livré.

- La commande est considérée comme « Livrée ».

A2. L'administrateur modifie le statut d'un produit d'une commande dont il en est le seul.

- A2.1. L'administrateur indique la fourniture comme « Livrée ».

La commande est considérée comme « Livrée ».

- A2.2. L'administrateur indique la fourniture comme « Rejetée ».

La commande est considérée comme « Rejetée ».

- A2.3. L'administrateur clique sur le bouton « Ne pas traiter cette commande ».

La commande est considérée comme « Rejetée ».

A3. L'administrateur modifie l'état d'une commande déjà indiquée comme « Livrée » ou « Rejetée ».

Le statut sera mis à jour par rapport à l'ancien statut de la commande (Livrée/Rejetée) ainsi que du nombre de fournitures présentes dans celle-ci.

5.7.3. 3eme cas d'utilisation : Exports Excel

Titre : Exports Excel

Résumé : L'administrateur a la possibilité d'exporter des fichiers Excel pour afficher les commandes des utilisateurs, une commande précise ou alors des statistiques des fournitures les plus vendues sur un temps donné.

Acteur : Administrateur

Création : 27/09/22

Date de mise à jour : 02/01/23

Responsable : Alizée HETT

Description des scénarios

Préconditions

- L'administrateur possède une connexion internet stable.
- L'administrateur est connecté à la plateforme.

Scénario nominal pour l'export d'une commande précise

1. L'administrateur se rend dans la partie administrateur, gestion des commandes
2. L'administrateur clique sur le bouton désignant Excel sur la commande qu'il veut exporter
3. Le fichier Excel se télécharge sur l'ordinateur de l'administrateur

Scénario nominal pour l'export de toutes les commandes d'un utilisateur précis

1. L'administrateur se rend dans la partie administrateur, gestion des utilisateurs
2. L'administrateur clique sur le bouton désignant Excel l'utilisateur dont il veut toutes les commandes
3. Le fichier Excel se télécharge sur l'ordinateur de l'administrateur

Scénario nominal pour l'export des fournitures vendues selon la date filtrée par ordre croissant

1. L'administrateur se rend dans la partie administrateur, statistique
2. L'administrateur clique sur la date voulue, courant du mois, sur 1an, sur 2ans et toutes les fournitures vendues.
3. Le fichier Excel se télécharge sur l'ordinateur de l'administrateur

6. Tests unitaires

- Liste des scripts relatifs aux tests unitaires et à l'analyse du code, disponible dans le fichier « package.json » :

```
"scripts": {  
  "tests": "php artisan test --stop-on-failure",  
  "phpunit": "./vendor/bin/phpunit --dont-report-useless-tests --stop-on-failure",  
  "phpstan": "./vendor/bin/phpstan analyse -c phpstan.neon --xdebug",  
  "sail:tests": "./vendor/bin/sail npm run tests",  
}
```

6.1. Lancement des tests :

```
npm run tests  
  
# Attention si vous utilisez docker, la commande sera la suivante :  
npm run sail:tests
```

6.2. Analyse du code :

- Utilisation de [PHPStan](#).

Attention : cette commande requiert l'installation de l'extension php [xdebug](#).

```
npm run phpstan
```

7. Configuration de l'envoi d'emails

7.1. Postfix

- L'utilisation de Postfix dans notre cas permet de diminuer la probabilité que nosemails soient considérés comme des spams.
- En utilisant Postfix en tant que relais SMTP cela fait en sorte que les emails ne partent pas en directement du serveur SMTP de google mais de notre serveur.

7.1.1. Installation

L'installation de Postfix se fait via la commande ci-dessous :

```
sudo apt install postfix
sudo apt-get install libsasl2-2 libsasl2-modules
```

7.1.2. Configuration

7.1.2.1. Authentification

- Créer le fichier suivant :

```
nano /etc/postfix/sasl_passwd
```

- Ajouter la ligne suivante dans le fichier précédemment créé :

Attention, cette configuration permet d'envoyer les mails vers le serveur smtp de Gmail.

```
[smtp.gmail.com]:587 user.name@gmail.com:password
```

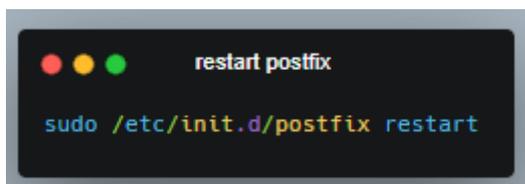
Générer la base de données d'authentification :

Attention, cette étape doit être réalisée à chaque modification du fichier «`/etc/postfix/sasl_passwd`».



```
postfix
postmap /etc/postfix/sasl_passwd
```

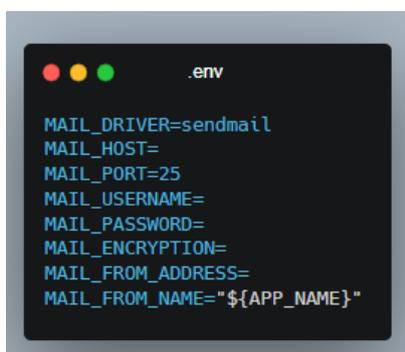
- Redémarrer postfix avec la commande suivante :



```
restart postfix
sudo /etc/init.d/postfix restart
```

7.1.2.2. Configuration de Laravel

Configuration du fichier «`.env`» de l'application web :



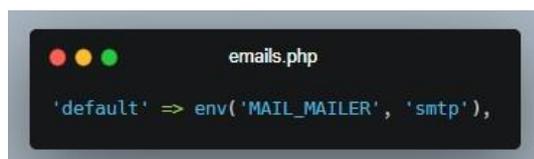
```
.env
MAIL_DRIVER=sendmail
MAIL_HOST=
MAIL_PORT=25
MAIL_USERNAME=
MAIL_PASSWORD=
MAIL_ENCRYPTION=
MAIL_FROM_ADDRESS=
MAIL_FROM_NAME="{APP_NAME}"
```

Ci-dessus, compléter les variables suivantes :

- «`MAIL_HOST`», par le nom d'hôte
- «`MAIL_PORT`», par le port attribué au protocole smtp
- «`MAIL_FROM_ADDRESS`», par l'adresse Gmail utilisée

Laravel récupère la valeur de la variable `MAIL_MAILER` du `.env` qui par défaut est `smtp`.

Cela fait qu'en indiquant `sendmail` comme driver, l'application va utiliser cette valeur.

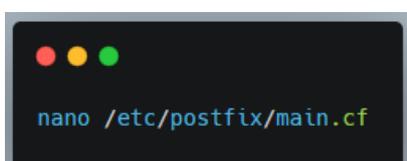


```
emails.php
'default' => env('MAIL_MAILER', 'smtp'),
```

7.1.2.3. Configuration de Postfix

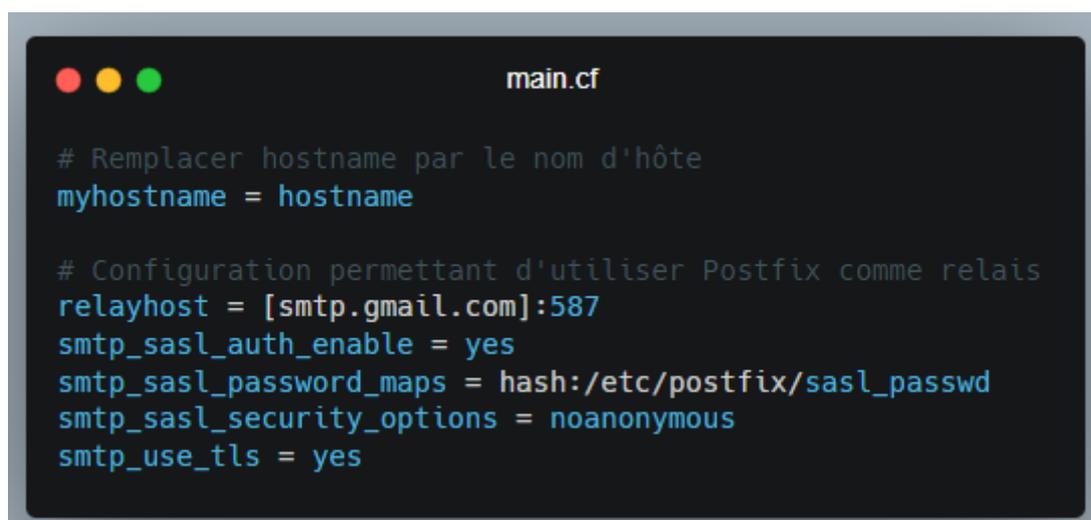
Afin de voir les modifications apportées au fichier de configuration vous pouvez le retrouver dans le répertoire docs : « docs\mails\postfix\conf\main.cf ».

Afin de modifier le fichier entrez la commande :



```
nano /etc/postfix/main.cf
```

Dans le fichier « main.cf », effectuer la configuration ci-dessous :



```
main.cf

# Remplacer hostname par le nom d'hôte
myhostname = hostname

# Configuration permettant d'utiliser Postfix comme relais
relayhost = [smtp.gmail.com]:587
smtp_sasl_auth_enable = yes
smtp_sasl_password_maps = hash:/etc/postfix/sasl_passwd
smtp_sasl_security_options = noanonymous
smtp_use_tls = yes
```

7.1.2.4. Changement du port du protocole SMTP

Le port par défaut du protocole smtp étant le port 25, il est nécessaire de le changer pour des raisons de sécurité.

Dans le fichier « /etc/postfix/master.cf », il vous suffira de changer « smtp », à gauche, par le port désiré.



```
smtp      inet  n       -       y       -       -       smtpd
```

La dernière étape est d'ouvrir le port sélectionné dans le pare-feu (dans notre cas, nous utilisons ufw) :



7.1.3. Éléments complémentaires

Accès aux logs :

- `/var/log/mail.err` : pour les erreurs du serveur
- `/var/log/mail.log` || `/var/log/mail.info` : envoi d'emails

Documentations complémentaires :

- [Documentation d'Ubuntu](#)
- [Configuration d'un relais smtp avec Postfix](#)
- [Configuration de Laravel pour utiliser postfix via Gmail smtp](#)
- [Installer et configurer un relais smtp avec postfix](#)

7.2. Supervisor

L'utilisation de supervisor nous permet d'effectuer l'envoi des emails en tâche de fond. Nous devons utiliser cela, car lors de l'envoi d'emails multiple, cela bloque l'utilisateur durant le temps des requêtes.

Son fonctionnement est assez simple : il enregistre les emails à envoyer dans une base de données nommé « jobs ». Une commande permet donc de lire la table et d'envoyer les emails correspondants. Cette commande nécessitant d'être lancé pour que l'envoi d'email soit réalisé, il faut alors un logiciel permettant de l'exécuter en tâche de fond régulièrement.

7.2.1. Installation

L'installation de Supervisor se fait via la commande ci-dessous :

```
# https://laravel.com/docs/9.x/queues#supervisor-configuration  
sudo apt-get install supervisor
```

7.2.2. Configuration

Après l'installation, il suffit de faire une légère configuration pour que cela fonctionne :

Rendez-vous dans le dossier de configuration de supervisor :

```
cd /etc/supervisor/conf.d
```

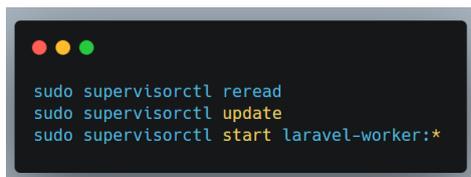
Créer un nouveau fichier correspondant à notre « worker » :

```
sudo nano laravel-worker.conf
```

Configurer le fichier avec le contenu suivant :

```
[program:laravel-worker]  
process_name=%(program_name)s_%(process_num)02d  
command=php /var/www/ccciappro/artisan queue:work sqs --sleep=3 --tries=3 --max-time=3600  
autostart=true  
autorestart=true  
stopasgroup=true  
killasgroup=true  
user=USER  
numprocs=8  
redirect_stderr=true  
stdout_logfile=LOGFILE  
stopwaitsecs=3600
```

Lancement du logiciel ainsi que du « worker » que nous avons créé :

A terminal window with a dark background and three colored window control buttons (red, yellow, green) in the top-left corner. The terminal displays three lines of text: 'sudo supervisorctl reread', 'sudo supervisorctl update', and 'sudo supervisorctl start laravel-worker:*'.

```
sudo supervisorctl reread
sudo supervisorctl update
sudo supervisorctl start laravel-worker:*
```

7.2.3. Éléments complémentaires

Erreurs liées aux droits :

- L'utilisateur effectuant la commande nécessite les droits sur le fichier **storage/logs/laravel.log**. Plusieurs façons de faire sont disponible :
- Documentations complémentaires :
 - [Documentation officielle de Laravel](#)