#### 

ENAC-IT2

EPFL | Nicolas Crausaz

Doc. Admin. – Canap-Gest

« Canap-gest » Application WEB de gestion de candidatures d’apprentissage epfl – Documentation administrateur

# Sommaire

[Sommaire 1](#_Toc8228926)

[1 Installation 2](#_Toc8228927)

[1.1 Dépendances en développement 2](#_Toc8228928)

[1.2 Dépendances en production 2](#_Toc8228929)

[2 Configuration 2](#_Toc8228930)

[2.1 Préparation à la production 2](#_Toc8228931)

[2.2 Structure 3](#_Toc8228932)

[2.3 Fichiers 4](#_Toc8228933)

[3 Déploiement 5](#_Toc8228934)

[3.1 5](#_Toc8228935)

# Installation

## Dépendances en développement

L’environnement de développement pour faire tourner l’application nécessite les dépendances suivantes :

* WampServer (avec PHP 7.2) (<http://www.wampserver.com/>)
* Composer (<https://getcomposer.org/>)
* Node.js (<https://nodejs.org/en/>)
* @vue/cli (<https://cli.vuejs.org/>)
* Git (<https://git-scm.com/>)

## Dépendances en production

Le serveur de production devra être équipé de :

* Service Web (IIS, apache etc.)
* PHP 7.2
* Composer

# Configuration

## Préparation à la production

Afin de préparer la production, certains éléments définis lors du développement doivent être modifiés.

Pour commencer, clonez al dernière version stable du projet en local :

C:\..\> git clone https://c4science.ch/source/canapgest.git

Au niveau de l’interface, il suffit uniquement de modifier la configuration de Axios afin d’utiliser l’url en production de l’API :

Il faut donc éditer le fichier *Site/src/plugins/axios.js*, et de modifier l’instance Axios ainsi :

const instance = axios.create({

headers: { 'Authorization': "Bearer " + localStorage.getItem('stored\_token') },

baseURL: 'https://canap-gest.epfl.ch/api'

})

Pour finir, optimiser et générer l’application Vue.js pour la production :

C:\..\> cd Site

C:\..\Site\> npm install

C:\..\Site\> npm run build

Le résultat se trouve comme indiqué dans le dossier « dist ».

Au niveau de l’API il faut modifier plusieurs éléments : tout d’abord, modifier l’URL de redirection Tequila.

Pour ce faire, éditer le fichier *API/app/http/Controllers/AuthController.php en remplaçant*

$oClient->SetApplicationURL('http://canap-gest-dev.local:8080');

*par*

$oClient->SetApplicationURL('https://canap-gest.epfl.ch');

Renommer ensuite le fichier *API/.env.example* en *.env*, puis remplir les champs ainsi :

APP\_ENV=local

APP\_DEBUG=false

APP\_KEY=<chaîne de caractère aléatoire>

APP\_TIMEZONE=UTC

DB\_CONNECTION=mysql

DB\_HOST=mysql-fac.epfl.ch

DB\_PORT=33003

DB\_DATABASE=canap\_db

DB\_USERNAME=<username>

DB\_PASSWORD=<password>

CACHE\_DRIVER=file

QUEUE\_DRIVER=sync

JWT\_SECRET==<chaîne de caractère aléatoire>

Les informations de connexion mySQL se trouvent dans le KeePass ENAC-IT2.

COMPOSER INSTALL

## Structure

La structure du site, depuis sa racine web doit être structuré de cette manière :

* api
* www

Le dossier api contient donc le contenu du dossier « API », le www contient les fichiers obtenus après avoir effectué le « build » de l’application Vue.js (dossier « dist »).

Le dossier www doit être configuré en tant que destination par défaut.

## Fichiers

Il faut ensuite ajouter quelques fichiers de configuration pour assurer le bon fonctionnement de l’application. Les exemples suivants sont valables pour une utilisation de IIS en tant que service Web.

Ajouter à la racine du dossier api, un fichier « web.config » contenant :

<?xml version="1.0" encoding="UTF-8"?>

<configuration>

<system.webServer>

<handlers>

<remove name="PHP\_7.2" />

<add name="PHP\_7.2" path="\*.php" verb="GET,HEAD,POST,DEBUG,PUT,DELETE,PATCH,OPTIONS" type="" modules="FastCgiModule" scriptProcessor="C:\Program Files (x86)\PHP\v7.2\php-cgi.exe" resourceType="Either" requireAccess="Script" allowPathInfo="false" preCondition="" responseBufferLimit="4194304" />

</handlers>

<rewrite>

<rules>

<rule name="Silex Front Controller" stopProcessing="true">

<match url="^(.\*)$" ignoreCase="false" />

<conditions logicalGrouping="MatchAll">

<add input="{REQUEST\_FILENAME}" matchType="IsFile" ignoreCase="false" negate="true" />

</conditions>

<action type="Rewrite" url="/api/public/index.php" appendQueryString="true" />

</rule>

</rules>

</rewrite>

</system.webServer>

</configuration>

Cette configuration va rediriger toutes les requêtes API vers le routeur Lumen.

Ajouter ensuite à la racine du dossier www, un fichier « web.config » contenant :

<?xml version="1.0" encoding="UTF-8"?>

<configuration>

<system.webServer>

        <modules runAllManagedModulesForAllRequests="true">

            <remove name="WebDAVModule" />

        </modules>

<defaultDocument enabled="true">

<files>

<clear />

<add value="index.html" />

</files>

</defaultDocument>

<rewrite>

<rules>

<clear />

<rule name="Redirect to https" stopProcessing="true">

<match url=".\*" />

<conditions>

<add input="{HTTPS}" pattern="off" ignoreCase="true" />

</conditions>

<action type="Redirect" url="https://{HTTP\_HOST}:8443{REQUEST\_URI}" redirectType="Permanent" appendQueryString="false" />

</rule>

</rules>

</rewrite>

<handlers>

<remove name="WebDAV" />

<add name="PHP\_7.2" path="\*.php" verb="GET,HEAD,POST" type="" modules="FastCgiModule" scriptProcessor="C:\Program Files (x86)\PHP\v7.2\php-cgi.exe" resourceType="Either" requireAccess="Script" allowPathInfo="false" preCondition="" responseBufferLimit="4194304" />

</handlers>

</system.webServer>

</configuration>

Cette deuxième configuration permet de servir l’application si l’URL n’est pas une requête vers l’API.

# Déploiement

## 

Ajouter création DB / User / Script