CORTEST
Prise en main

Jean-Baptiste Moussa, Julien Collard, Louis Proffit

April 12, 2023

1 Description de l'environnement

L'application Cortest est une application web PHP basée sur le framework Symfony. Son environnement est composé de :

1. Un exécutable **PHP**, en version **8.1 minimum**
2. Le système de gestion de build **composer**, en version **2.5 minimum**.
3. Une base de données SQL. Ex : **MySQL**, **PostgreSQL**.
4. Un serveur web PHP. Ex: **Apache**, **Nginx**.
5. Un compilateur Latex. Ex : **pdflatex**.

2 Déploiement

Avant de déployer le projet Cortest, il faut récupérer les fichiers sources. Ceux-ci sont disponibles sur le système de gestion de version **git** à l'url https://github.com/Louis-Proffit/Cortest.git. La branche **main** par défaut est la bonne.
La commande suivante permet de récupérer les fichiers sources :

```
$ git clone https://github.com/Louis-Proffit/Cortest.git
```

Une fois les fichiers sources récupérés, il faut déployer l'application.
Deux options existent pour déployer Cortest :

- Pur, avec des dépendances externes
- Sur une VM (Docker)

2.1 Déploiement pur

Pour un déploiement pur, les dépendances doivent être installées manuellement et séparément.

L'exécutable PHP, dans la bonne version, doit être accessible sur la ligne de commande pour simplifier les autres procédures.

Composer peut-être accessible sur la ligne de commande, ou sous la forme d'une archive **.phar**, à la racine. L'archive est accessible à l'URL : https://getcomposer.org/download/latest-stable/composer.phar.
La commande suivante télécharge l'archive composer

```
$ wget https://getcomposer.org/download/latest-stable/composer.phar
```

Un système de gestion de base de données SQL doit être accessible par l'application. Il est possible d'en installer un uniquement pour Cortest, ou d'en utiliser un existant en créant un utilisateur ayant accès à au moins une base de données.

La base de données doit être accessible à une URL, qu'il faudra ajouter en variable d'environnement dans la configuration de Cortest (voir 2.3 pour les variables d'environnement).

La racine du serveur web n'est pas la racine du projet mais le dossier **./public**. Le serveur web peut prendre plusieurs formes :

- Le serveur web intégré de Symfony. Les instructions d'installation sont disponibles à l'url https://symfony.com/download. Une fois la commande *symfony* disponible, la commande

  symfony server:start

  permet de lancer un serveur web déjà configuré. Entre autres, cette commande doit être exécutée à la racine du projet et non pas dans le dossier **public** (elle fait le changement elle-même).

- Un serveur web Apache. Les instructions d'installation sont disponibles à l'url https://httpd.apache.org/download.cgi. Les instructions de lancement du serveur sont disponibles à l'url https://httpd.apache.org/docs/current/fr/. Le fichier de configuration du serveur Apache (**.htaccess**), est déjà rédigé sur le chemin **./public/.htaccess**. Le serveur Apache doit être exécuté avec pour racine le dossier **public**.

Un compilateur Latex doit être disponible dans la ligne de commande du serveur web. Par défaut, Cortest attend le compilateur **pdflatex**. Si un autre compilateur est installé (non recommandé), il faudra le préciser dans les fichiers de configuration (non détaillé ici, nous contacter).

La commande suivante permet d'installer un compilateur Latex fonctionnel.

```
$ apt-get -y --no-install-recommends install texlive-latex-base \
  texlive-fonts-recommended  \
  texlive-fonts-extra \
  texlive-latex-extra
```

L'installation peut être testée avec la commande *pdflatex*, qui doit afficher la version du compilateur.

2.2 Déploiement sur VM (Docker)

Les dépendances ont été conteneurisées dans les fichiers **./docker-compose.yml** et **./Dockerfile**. Le premier spécifie l'environnement souhaité, et le second précise la configuration du serveur web, y compris l'installation de **composer**, de **php** et du compilateur **LateX**.

Le système de base de données utilisé est **MySQL**. Le serveur web est **Apache**, le compilateur Latex est **pdflatex**.

Pour déployer grâce à Docker, il faut disposer de Docker Desktop (Windows, Linux, MacOS...). Les instructions d'installation sont disponibles à l'url suivante : https://www.docker.com/get-started/. Une fois installé, la fonctionnalité **Docker compose** doit être activée. Les instructions pour cela sont disponibles à l'url https://docs.docker.com/compose/install/.

Docker doit tourner en arrière-plan pour que ses conteneurs puissent fonctionner.
Quand c'est le cas, la commande suivante permet d'exécuter le conteneur :

```
$ docker-compose up --build
```

Cortest sera accessible sur le port 8080 de la machine (typiquement **localhost:8080**).

2.3 Parties communes au déploiement

La base de données SQL doit être synchronisée avec l'application Cortest. De plus, certaines données par défaut doivent être insérées (au moins à l'initialisation) dans la base de données.

Pour synchroniser le schéma SQL entre la base de données et l'application, deux options s'offrent :

- La synchronisation de force, quitte à supprimer le contenu de la base de données qui ne correspond pas au nouveau schéma. C'est l'option à choisir lors de l'initialisation, puisque aucune donnée ne peut être supprimée. La commande à effectuer est :

  ```
  $ php bin/console doctrine:schema:update --force
  ```

  Ne pas exécuter cette commande quand la base de données contient des valeurs, ou alors après avoir sauvegardé toutes les données importantes.

- La migration. Le processus est plus complexe, et est détaillé à l'url suivante : https://symfony.com/bundles/DoctrineMigrationsBundle/current/index.html. La migration ne supprime aucune donnée mais nécessite un travail d'adaptation "à la main" au cas par cas. A utiliser en cours de déploiement.

Une fois le schéma SQL initialisé, certaines lignes de la base de données "par défaut" doivent être insérées (Niveaux scolaires, valeurs de test, certains correcteurs / etalonnages / graphiques...). Pour cela, la commande à effectuer est :

```
$ php bin/console doctrine:fixtures:load
```

Il est demandé une confirmation, puisque le chargement de ces lignes écrasera le contenu actuel de la base de données. Cette commande ne doit donc être effectuée qu'après une initialisation, ou après la sauvegarde de toutes les données importantes.

Les commandes précédentes doivent être exécutées à la racine du projet (le chemin **bin/console** est relatif à cette racine). Dans le cas d'un déploiement par Docker, la racine du projet en question est celle de la machine virtuel, pas du répertoire d'ou le git a été cloné. Il faut donc ouvrir un terminal et se placer dans le dossier **/var/www** qui est la nouvelle racine. Cela peut se faire directement depuis l'application Docker Desktop, ou grâce à la commande

```
$ docker exec -it [process_id] bash
```

L'identifiant du processus peut être trouvé grâce à la commande

```
$ docker ps
```

3 Côté client

Les postes clients de l'application doivent être équipés de **Chrome** ou **Chromium** pour bénéficier du module **Serial Port** de Java Script.

La connexion au lecteur optique se fait via un adaptateur Serial Port vers USB branché sur un port USB. Au branchement de l'appareil, le périphérique doit être reconnu et interprété comme un Serial Port. Dans le gestionnaire de périphérique windows, il doit apparaître comme **USB Serial Port (COM5)** dans la catégorie **Ports (COM et LPT)**. Si ce n'est pas le cas et qu'il apparaît dans la catégorie **Autres périphériques**, désinstaller l'appareil et le rebrancher peut solutionner le problème. Si ce dernier persiste, il manque certainement un pilote sur le poste.

L'utilisation de l'application se fait ensuite depuis le navigateur Chrome ou Chromium.