

# Mon Projet CIEL-2 (GTB ou CO)

platform MacOS | Linux | Windows

License MIT

Python 3.9+

PHP &gt;=7.4

MySQL &gt;=5.7

JavaScript ES6+

Status En développement

repo size 27.4 MiB

issues 0 open







pull requests 0 open

last commit april

build passing

Un projet modulaire pour tester, analyser et développer des solutions innovantes.

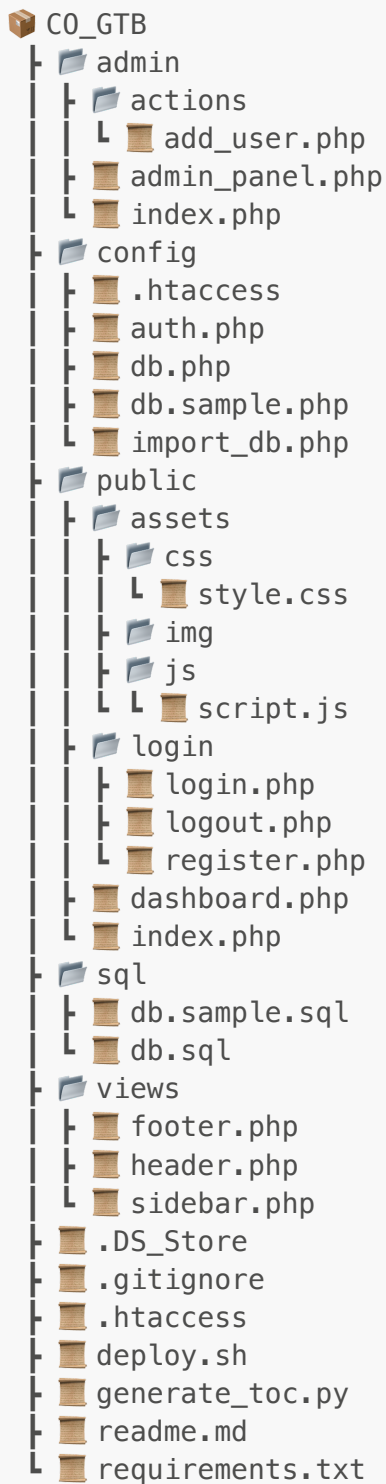
## Sommaire

- Mon Projet CIEL-2 (GTB ou CO)
  -  Sommaire
  -  Objectifs
  -  Structure du projet
  - Gestion des fichiers temporaires et sensibles
    - 1. Fichiers **.DS\_Store** à supprimer (Pour ceux qui travaillent avec un Mac)
    - 2. Fichier **.gitignore**
  -  3. Sécurité avec **.htaccess**
    - Dans **config/**
    - À la racine
  - 4. Déploiement avec (**deploy.sh**)
    - ➤ Utilisation
    - Script **deploy.sh** (à lancer depuis la racine)
  - Installation
  - Contribution
    - Création du dépôt GitHub
      - À venir
  - Licence
  -  Bonnes pratiques
  - Automatisation du sommaire
    -  Utilisation

## Objectifs

- Concevoir des environnements modulaires et reproductibles.
- Explorer et intégrer des approches innovantes.
- Offrir un cadre de prototypage rapide, configurable et évolutif.

## Structure du projet



---

## Gestion des fichiers temporaires et sensibles

### 1. Fichiers **.DS\_Store** à supprimer (Pour ceux qui travaillent avec un Mac)

Les fichiers **.DS\_Store** sont générés par macOS et n'ont aucune utilité dans le projet. Pour les supprimer et les ignorer dans Git :

```
find . -name ".DS_Store" -delete
```

Et ajouter dans **.gitignore** :

```
.DS_Store
```

---

## 2. Fichier **.gitignore**

Le fichier **.gitignore** permet d'exclure du dépôt Git les fichiers sensibles ou inutiles (ex : **config/db.php**, **.DS\_Store**, fichiers de logs, etc.).

Exemple à placer à la racine du projet :

```
# Fichiers systèmes
.DS_Store
Thumbs.db

# Environnements d'IDE
.vscode/
.idea/

# Dossiers sensibles ou inutiles pour le dépôt
logs/
dist/

# Fichiers de config personnels
config/db.php
config/*.local.php

# Fichiers temporaires
*.log
*.bak
*.tmp
*.swp

# Dump SQL local
*.sql

# Fichiers de déploiement
deploy.sh
```

---

## 3. Sécurité avec **.htaccess**

Dans **config/**

Bloque l'accès aux fichiers sensibles :

```
<FilesMatch "\.(php|ini|sql|bak)$">
    Order allow,deny
    Deny from all
</FilesMatch>
```

À la racine

Désactive l'indexation automatique des répertoires :

```
Options -Indexes
```

---

## 4. Déploiement avec (deploy.sh)

Le dossier `dist/` généré par le script `deploy.sh` sert à contenir la version Le script `deploy.sh` permet de préparer une version propre et sécurisée du projet pour la production.

- Crée un dossier `dist/` contenant uniquement les fichiers nécessaires.
- Ignore les fichiers sensibles ou inutiles.
- Peut compresser en `.tar.gz` le dossier pour faciliter le transfert.

### ► Utilisation

Rendre le script exécutable :

```
chmod +x deploy.sh
```

Lancer le script :

```
./deploy.sh
```

---

### Résultat

Type de fichier	Contenu réel	Version suivie par Git	Version visible en prod
<code>config/db.php</code>	mot de passe réel	✗ non (via <code>.gitignore</code> )	✓ oui (en ligne)
<code>config/db.sample.php</code>	structure vide	✓ oui	✗ non (juste pour dev)
<code>.DS_Store, *.log</code>	bruit système	✗ non	✗ non
<code>.htaccess</code>	règles de sécurité	✓ oui	✓ oui (active sur Apache)

---

## Script `deploy.sh` (à lancer depuis la racine)

Ce script :

- Crée un dossier `dist/`
- Copie les fichiers utiles
- Ignore les fichiers sensibles ou inutiles
- Peut être enrichi pour envoyer le tout en FTP ou SSH

```
#!/bin/bash

echo "Déploiement en cours..."

# Vérification de la présence de rsync
if ! command -v rsync &> /dev/null
then
    echo "Erreur : rsync n'est pas installé. Veuillez l'installer pour continuer."
    exit 1
fi

# Nettoyage du dossier précédent
if [ -d "dist" ]; then
    rm -rf dist || { echo "Erreur : Impossible de supprimer l'ancien dossier dist."; exit 1; }
fi
mkdir dist || { echo "Erreur : Impossible de créer le dossier dist."; exit 1; }

# Copie des fichiers principaux (en respectant les exclusions)
rsync -av --exclude='.git' \
        --exclude='.DS_Store' \
        --exclude='*.log' \
        --exclude='config/db.php' \
        --exclude='deploy.sh' \
        ./ dist/ || { echo "Erreur : Échec de la copie des fichiers."; exit 1; }

echo "Fichiers copiés dans /dist"

# Ajouter éventuellement une version prod de .htaccess
cp .htaccess dist/.htaccess || { echo "Erreur : Impossible de copier le fichier .htaccess."; exit 1; }

# (Optionnel) Compression du dossier dist
tar -czvf dist.tar.gz dist/ || { echo "Erreur : Impossible de compresser le dossier dist."; exit 1; }

echo "📁 Structure de prod prête dans /dist"
echo "Dossier compressé en dist.tar.gz"
echo "On peut maintenant envoyer le dossier via scp ou FTP."
```

On rend ce fichier exécutable avec :

```
chmod +x deploy.sh
```

Et on le lance avec :

```
./deploy.sh
```

---

## Installation

1. Cloner le dépôt et installer les dépendances :

```
git clone https://github.com/MonGitHub/CO_GTB.git
cd CO_GTB
pip install -r requirements.txt
```

2. Configurer la base de données :

- Importer le fichier `db.sql` dans la base de données MySQL.
- Modifier le fichier `config/db.php` avec les vraies informations de connexion.

3. Ou

- Copier `db.sample.php` → `db.php` et renseigner vos accès MySQL
- Puis lancer :

```
php config/import_db.php
```

Cela importe `db.sample.sql` dans la base.

Ces scripts facilitent l'expérimentation sans configuration complexe.

---

## Contribution

Les contributions sont les bienvenues ! Veuillez soumettre une *pull request* ou ouvrir une *issue* pour signaler des problèmes.

1. **Fork** le repo
2. Crée une nouvelle branche : `git checkout -b ma-feature`
3. Fais tes modifications (code, docs, tests...)
4. Commit : `git commit -m "Ajout de ma feature importante"`

5. Push ta branche : `git push origin ma-feature`

6. Ouvre une **pull request**

## Création du dépôt GitHub

Initialiser le dépôt et le publier sur GitHub :

```
git init
git add .
git commit -m "Initial commit"
gh repo create mon_projet --public --source=. --remote=origin
git push -u origin main
```

## À venir

- ☐ Ajout d'une base de données (MySQL)
- ☐ Authentification inter-services
- ☐ Intégration de tests unitaires (PHPUnit / pytest)

---

## Licence

Ce projet est sous licence MIT. Consultez le fichier `LICENSE` pour plus d'informations.

---

## Bonnes pratiques

- **Ne jamais versionner les mots de passe ou secrets** : utilisez des fichiers d'exemple (`db.sample.php`).
  - **Nettoyez régulièrement les fichiers temporaires.**
  - **Protégez les dossiers sensibles avec `.htaccess`.**
  - **Utilisez le script de déploiement pour éviter d'exposer des fichiers inutiles ou sensibles en production.**
- 

## Automatisation du sommaire

### Utilisation

1. Enregistrer le fichier sous `generate_toc.py` dans le même dossier que le `readme.md`. (voir fichier `joint`)
2. Lancer-le avec :

```
python generate_toc.py
```

3. S'assurer d'avoir dans le `readme.md` une section avec les balises :

```
<!-- TOC START -->  
<!-- TOC END -->
```