

MARIE-ANDRÉE HEALEY-CÔTÉ

Programmation Web avancée

582-31B-MA, gr.24610

TP 3

**Architecture MVC – Administrateur - Client**

Travail présenté à

M. Marcos Sanches

Département de Formation continue – Programme NWE.OF

Collège de Maisonneuve

Le 17 novembre 2025

# Présentation du projet : Interface Administrateur et Client (MVC – OOPHP)

## 1. Contexte et objectifs

Ce projet a pour objectif de développer un site Web dynamique en PHP en appliquant une **architecture MVC orientée objet**, garantissant une structure claire, évolutive et bien organisée. Le site intègre deux espaces distincts :

- **Une interface client**, destinée à la consultation du contenu,
- **Une interface administrateur**, permettant la gestion complète du site.

Afin d'assurer une utilisation sécurisée, une base de données enrichie a été mise en place pour gérer les utilisateurs, leurs rôles et leurs droits d'accès. Les mots de passe sont **chiffrés**, et des **sessions sécurisées** sont utilisées pour authentifier les visiteurs et contrôler l'accès aux pages sensibles.

L'administrateur bénéficie d'un tableau de bord lui permettant d'**ajouter ou de modifier le contenu du site**, tandis que l'activité des visiteurs est enregistrée dans un **journal de bord** comprenant l'adresse IP, la date, le nom d'utilisateur (ou "visiteur") et la page consultée. Ce journal est accessible via le menu de navigation.

Pour enrichir l'expérience utilisateur, le projet intègre également une **fonctionnalité additionnelle** au choix, comme l'envoi d'e-mails, la génération de fichiers PDF, le téléversement d'images ou la gestion multilingue.

Ce projet met en pratique les principes de la programmation orientée objet, la sécurité Web, la gestion de base de données et la création d'interfaces professionnelles.

## 2. Architecture et organisation

L'application sera structurée selon le modèle **MVC** :

- **Modèle (Model)** : gère les interactions avec la base de données et représente les entités du système.
- **Vue (View)** : s'occupe de l'affichage et du rendu des données via le moteur de template **Twig**.
- **Contrôleur (Controller)** : traite les requêtes, appelle les modèles, et envoie les données aux vues.

Une gestion claire des routes sera mise en place pour faire le lien entre les URL et les contrôleurs.

## 2.1 Architecture du projet

mvc/

controllers/ ← Dossier contenant les contrôleurs

HomeController.php

LivreController.php

AuthController.php

UserController.php

LogController.php

models/ ← Dossier contenant les modèles

CRUD.php

Livre.php

Categorie.php

Auteur.php

Editeur.php

ExampleModel.php

Privilege.php

User.php

Log.php

providers/ ← Dossier contenant les "providers"

Validator.php

View.php

Auth.php

public/ ← Dossier contenant tout ce qui est domaine public

css/

style.css ← Feuille de style du site

img/ ← images utilisées sur le site

uploads/ ← images uploadées

routes/            ← Dossier contenant les routes

Route.php

web.php

views/            ← Dossier contenant les vues

auth/

create.php

index.php

layouts/

header.php      ← Entête de page (composante)

footer.php      ← Pied de page (composante)

livre/

index.php        ← Page principale

create.php        ← Formulaire d'ajout de livres

edit.php          ← Formulaire de modification de livres

show.php         ← Script qui récupère les détails du livre (bd) et affiche les infos

user/

create.php        ← Formulaire d'inscription d'un utilisateur (accessible admin)

log/

show.php         ← Journal de bord (pages visitées)

error.php

home.php

.htaccess

composer.json

composer.lock

config.php

index.php

### 3. Développement

Depuis le TP2 :

- Ajouter les tables et les champs nécessaires dans la base de données pour pouvoir gérer les utilisateurs.
- Les mots de passe doivent être chiffrés.
- Créer une session sécurisée pour les clients et les administrateurs.
- Gérer les privilèges d'accès aux pages
- L'administrateur doit pouvoir ajouter du contenu au site à l'aide de l'interface.
- Gérer la connexion de l'utilisateur, le journal de bord, avec l'adresse IP, la date, le nom d'utilisateur (si l'utilisateur est connecté, sinon s'inscrire en tant que visiteur) et la page visitée.
- Le journal de bord doit être accessible à partir du menu de navigation du site.
- Ajouter une nouvelle fonctionnalité de mon choix :
  - Téléverser (upload) des images et enregistrer dans la base de données pour les publier sur le site Web.

\* À noter que cette fonctionnalité sera disponible lors d'ajout de livres ou modification de livre. Il sera possible d'ajouter (create) ou de modifier (edit) une image (couverture de livre dans le cas présent) via le formulaire. (Note : gérer par le LivreController)

### 4. Conclusion

Tel que mentionné dans l'intro, ce projet met en pratique les principes de la programmation orientée objet (OOPHP), la sécurité Web, la gestion de base de données et la création d'interfaces professionnelles.

### Lien GitHub

Lien : <https://github.com/marieAndreeHealeyCote/TP03>

### Utilisateurs

Utilisateur : [admin@test.ca](mailto:admin@test.ca)

Mot de passe : 123456

Utilisateur : [client@test.ca](mailto:client@test.ca)

Mot de passe : 123456

## Structure de la base de données (ERD)

