# Organisation du TP

Ce TP va se dérouler en plusieurs séances, et va demander beaucoup de travail ! Pour ce faire, vous allez vous organiser en **équipes projet** de 2 ou 3 personnes **maximum**.

Puisque ce projet doit être collaboratif, votre équipe doit créer un unique dépôt Git qui contiendra votre application.

Pour initialiser le projet, je vous recommande de faire le processus de mise en place sur 1 seul poste, puis de créer un dépôt Git et de pousser le code du projet dans votre dépôt. Les autres membres de votre équipe projet n’auront plus qu’à cloner le projet Git et installer les dépendances Composer pour pouvoir travailler sur le projet (voir ci-dessous).

# Contexte

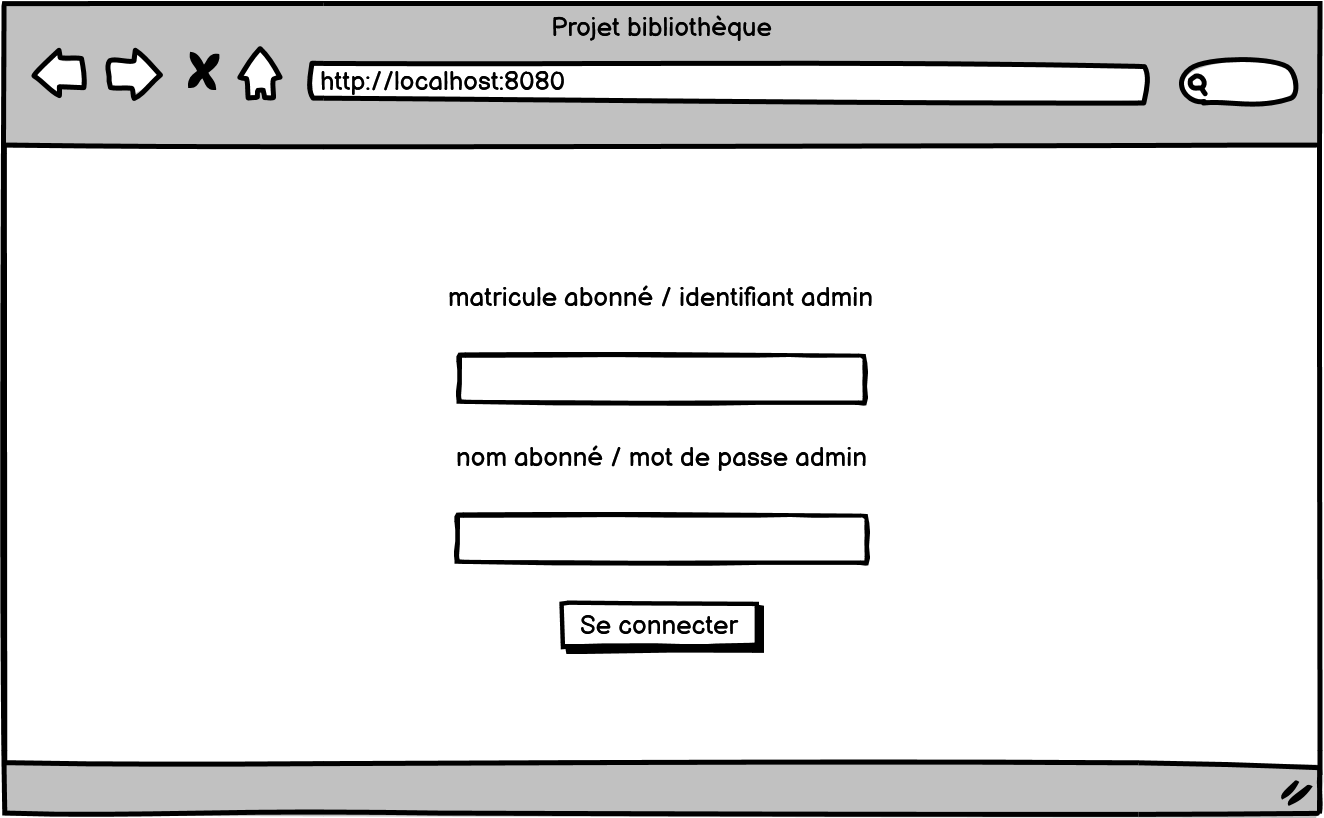
Faisant suite au TP2 de base de données, nous allons implémenter ensemble le logiciel de gestion de bibliothèques. Pour ce faire, nous allons développer l’application comme une application web qui s’appuiera sur le Design Pattern Model-View-Controller et qui va utiliser le framework CodeIgniter.

Le modèle de données ayant été corrigé en cours, je fournis joint à ce sujet le script SQL de création de la base de données suivant : [TP3 - script creation SGBD.sql](https://drive.google.com/file/d/15LZR5WI09konkjX3dMvffXPKsGXNxQkh/view?usp=sharing)

Le but de ce TP est de développer l’application de gestion de bibliothèques en plusieurs étapes. **Développer le projet en suivant les étapes !**

## Etape 1 : Connexion & Espace administrateur

Dans un premier temps, créer une première route “/login” qui va permettre d’afficher un formulaire de connexion comme modélisé dans la maquette ci-dessous :

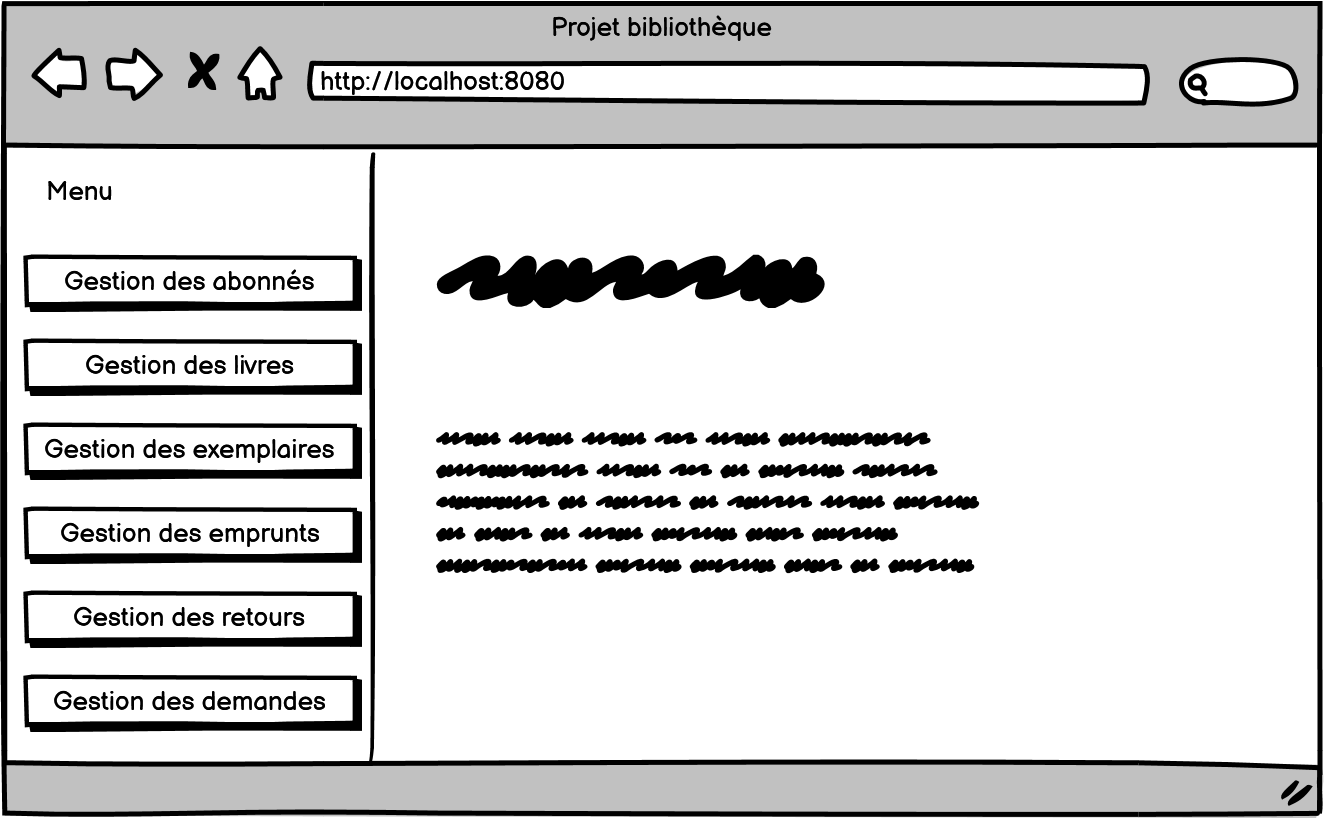


La page demande à se connecter à l’aide d’un login / mot de passe. Nous allons enregistrer dans les constantes de l’app CodeIgniter le login / mot de passe administrateur afin de ne pas gérer l’authentification en base de données.

La requête sur /attempt-login va soumettre le formulaire et valider que les infos soient OK pour authentifier et connecter l’administrateur.

Il faudra mettre en place une sécurité qui empêche qu’un utilisateur non connecté ne puisse pas accéder aux autres pages !

Une fois connecté, une page d’accueil apparaît :



l’interface administrateur doit permettre :

* d’ajouter un Livre à la bibliothèque (et de créer l’auteur et les mot-clés le cas échéant) à l’aide d’un formulaire.
* de consulter la liste des livres
* d’ajouter un exemplaire d’un livre existant
* de consulter la liste des exemplaires
* d’enregistrer un abonné
* de consulter les informations des abonnés sous forme de tableau
* de consulter les informations d’un abonné (et de modifier ces informations)

## Etape 2 : Espace Abonné

Créer une interface abonné qui demande à se connecter à l’aide de son matricule et de son nom. Une fois connecté, cette interface doit permettre :

* de consulter la liste des livres (réutiliser le code existant !)
* de consulter la liste des exemplaires disponibles à l’emprunt (réutiliser le code existant !)
* de consulter la liste des exemplaires empruntés par l’abonné connecté
* de consulter les informations de l’abonné connecté (et de modifier ces informations le cas échéant, à l’exception de son matricule abonné) (réutiliser le code existant !)

## Etape 3 : Gestion des Emprunts

Améliorer l’interface d’administration afin de pouvoir enregistrer des emprunts. On doit désigner un abonné, un exemplaire disponible, et enregistrer l’emprunt à la date du jour.

L’interface d’administration doit également pouvoir consulter la liste des emprunts actifs, c’est-à-dire qu’il doit afficher la liste des emprunts (le matricule / nom de l’abonné, la cote de l’exemplaire, le code catalogue et le titre du livre, la date d’emprunt, la date de retour, et si l’abonné a demandé le renouvellement de son emprunt) sous forme de tableau.

Enfin l’interface d’administration doit permettre de rendre un livre en saisissant la cote de l’exemplaire, vérifier si l’exemplaire est effectivement emprunté, et supprimer la ligne d’emprunt correspondante.

## Etape 4 : Gestion des demandes

De la même manière que pour la gestion des emprunts, créer dans l’interface utilisateur une page de création d’une demande de livre. Cette page doit vérifier que le livre n’est pas disponible (qu’il existe au moins un exemplaire non emprunté) et afficher son emplacement si besoin. Dans le cas contraire, on enregistre une demande d’emprunt à la date du jour.

On créera également dans l’interface Abonné une page “Mes demandes” qui liste les demandes d’emprunts de l’abonné, et qui permet pour chaque ligne de supprimer la demande (si l’abonné n’a plus envie d’emprunter ce livre).

Dans l’interface administrateur, rajouter une pages gestion des demandes qui va afficher la liste des demandes, par livre puis par date de demande croissante afin d’afficher une priorité.

Pour chaque ligne, l’administrateur peut supprimer la demande d’emprunt.

## Etape 5 : Gestion de l’usure

Dans l’interface administrateur, modifier la gestion des retours d’emprunts pour être en mesure d’indiquer l’état d’usure des exemplaires.

On décide arbitrairement que les états possibles de l’usure d’un livre sont les suivants :

* NEUF
* TRES BON
* BON
* MOYEN
* DEGRADE

On décide également de ne plus afficher dans la vue de gestion des exemplaires et dans la gestion des emprunts de ne plus pouvoir voir les exemplaires qui sont à l’état **DEGRADE**.

On va également ajouter une page “Gestion de l’état des exemplaires” qui va afficher, pour chaque livre du catalogue de l’application, le nombre d’exemplaires de chaque état.

On veut afficher par livre et au total le pourcentage par état afin de pouvoir décider quels exemplaires racheter.

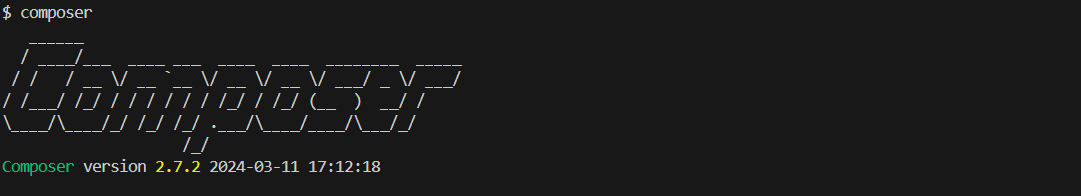
# Mise en place

## Initialisation du projet

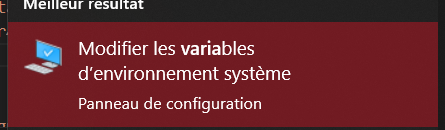
Dans un premier temps, il va vous falloir installer et configurer le framework **Code Igniter**. Pour ce faire, nous allons utiliser le gestionnaire de dépendances **Composer**, qui permet de télécharger et d’installer des dépendances PHP.

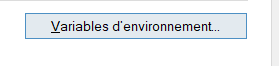
1. Suivre les instructions pour installer Composer ici : [Introduction - Composer](https://getcomposer.org/doc/00-intro.md)

Attention à choisir la bonne version de PHP (la plus avancée) !

1. Une fois Composer installé, vous pouvez valider son fonctionnement à l’aide de la commande composer :   
   
2. Pour créer un nouveau projet CodeIgniter à l’aide de Composer, suivre les instructions de [Composer Installation — CodeIgniter 4.4.5 documentation](https://www.codeigniter.com/user_guide/installation/installing_composer.html)
3. Se positionner dans le répertoire dans lequel vous voulez créer votre projet (C:/projects/TP3-bibliothèque par exemple) et lancer la commande :   
   composer create-project codeigniter4/appstarter .
4. Le framework **CodeIgniter** embarque un serveur spark de développement, il n’y a donc pas besoin de créer un virtual host avec wamp ! Une fois votre installation terminée, lancer le projet à l’aide de la commande php **spark serve**

💡 Si la commande **php** échoue, pas de panique ! Cela veut dire que votre fichier exécutable php.exe n’est pas défini dans la variable d’environnement **PATH** de votre système. Pour corriger le problème, suivre ces étapes :

* Sur Windows : ouvrir la **Gestion des Variables d’environnement** :   
  



Ajouter une entrée qui pointe vers l’exécutable PHP présent dans Wamp :   
  
Puis valider en cliquant sur OK, relancer un terminal et retester la commande php -version

* Sur Linux / Mac : à partir du terminal, créer un fichier texte .bash\_profile dans votre répertoire personnel à l’aide de la commande **nano ~/.bash\_profile** et y coller le texte suivant en remplaçant la version de PHP par celle installée par LAMP / MAMP :

| **export MAMP\_PHP=/Applications/MAMP/bin/php/<VERSION DE PHP>/bin**  **export PATH="$MAMP\_PHP:$PATH"** |
| --- |

Sauvegarder et relancer votre terminal, la commande **php -version**

## Récupération du projet

Cloner le projet à partir d’un dépôt Git, puis à partir du répertoire du projet, exécuter la commande **composer install** afin d’installer les dépendances du projet.

💡 A l’instar des modules npm contenus dans un dossier **node\_modules**, on ne pousse jamais les dépendances de composer car ils dépendent de votre machine, et on peut les retrouver à l’aide de la commande !