

**DOSSIER DE PROJET**

**Lionel ENSFELDER**

***Année 2019***

****

***Titre professionnel de niveau 5***

**Table des matières**

[Chapitre 1 – Conception d’une application web ou web mobile 3](#_Toc25934999)

[1. Liste des compétences du référentiel couvertes par le projet 3](#_Toc25935000)

[2. Résumé du projet 3](#_Toc25935001)

[3. Cahier des charges, expression des besoins et spécifications fonctionnelles du projet 4](#_Toc25935002)

[4. Spécifications techniques du projet 11](#_Toc25935003)

[Chapitre 2 – Présentation des éléments de réalisation 22](#_Toc25935004)

[1. Fonctionnement globale de l’application 22](#_Toc25935005)

[2. Tests unitaires 28](#_Toc25935006)

[Chapitre 3 – Description et mise en œuvre des compétences transversales 31](#_Toc25935007)

[1. Description de la veille sur les vulnérabilités de sécurité liée au développement d’une interface utilisateur web dynamique 31](#_Toc25935008)

[2. Description d’une situation ayant nécessité une recherche à partir d’un site anglophone 32](#_Toc25935009)

[3. Extrait du site anglophone utilisé pour la recherche décrite précédemment 33](#_Toc25935010)

# Chapitre 1 – Conception d’une application web ou web mobile

## Liste des compétences du référentiel couvertes par le projet

Les compétences du référentiel couvertes par le projet sont :

1. **Développer la partie Front-End d’une application web ou web mobile en intégrant les recommandations de sécurité :**

* Maquetter une application
* Réaliser une interface utilisateur web statique et adaptable
* Développer une interface utilisateur web dynamique

1. **Développer la partie Back-End d’une application web ou web mobile en intégrant les recommandations de sécurité :**

* Créer une base de données
* Développer les composants d’accès aux données
* Développer la partie Back-End d’une application web ou web mobile

## Résumé du projet

Movies Space est une interface web de gestion de films qui se base sur les données rentrées par l’utilisateur. Cette application est destinée dans un premier temps à rentrer les données d'un film manuellement via un formulaire puis les stocker en base de données.

La restitution de tous les films présents en base de données et des informations associées se fait via la page d'accueil dans le cas où l’utilisateur est connecté. Tous les utilisateurs connectés ont la possibilité de modifier les informations de tous les films présents en base de données. Ils pourront ainsi entrer et mettre à jour pour chaque film le titre, l’année, le tag associé, le(s) acteur(s), le réalisateur et la jaquette. Un utilisateur peut également supprimer le film de son choix.

Lorsqu’un utilisateur n’est pas connecté il ne peut accéder qu’aux pages de création de compte et de connexion. Un utilisateur connecté peut accéder aux fonctionnalités de création, modification et suppression d’un film.

Pour ce projet j’ai choisi d’utiliser HTML 5, CSS 3, Bootstrap 4, Twig pour le templating et JavaScript pour la partie Front-End. La partie Back-End, quant à elle, est construite autour du stack Apache, PHP, MySQL (uWamp sur clé USB) et comporte aussi du JavaScript. La gestion du projet s’est faite en utilisant la démarche Agile, le Framework Scrum ainsi que la méthode Kanban. Cette combinaison m’a permis de planifier mon projet en respectant des conventions et méthodologies efficaces utilisées par les professionnels partout dans le monde.

Cette application est également implémentée en Programmation Orientée Objet (POO).

Ce projet me permettra à terme de comprendre plus facilement les choix d'implémentation lors de l'abord d’un nouveau Framework et d’ajouter un niveau d’abstraction à mon style de programmation.

## Cahier des charges, expression des besoins et spécifications fonctionnelles du projet

1. **Contexte**

En tant que geek passionné de technologies et gros consommateur de films et de séries j’ai depuis longtemps envie d’avoir un serveur domestique sur lequel je ferai tourner des applications web accessibles par l’intermédiaire d’un Dashboard.

Ce tableau de bord serait matérialisé par une tablette me permettant d’avoir facilement accès à diverses informations comme la météo, mon calendrier, mes mails, l’état du trafic routier, mes playlists Spotify, ou encore une liste de films à voir.

Cette application web est donc un exercice préparatoire à la réalisation de ce projet personnel tout en satisfaisant aux besoins de la certification.

1. **Cible**

La cible principale de cette application est moi-même ainsi que les personnes présentes à mon domicile ayant un intérêt pour les médias en particulier les films. Ces personnes sont désireuses de garder à porter de main des informations sur des films à voir et pouvoir les modifier en cas de besoin. La simplicité de l’interface et un choix réduit de fonctionnalités sont très importants car la cible doit s’approprier l’application rapidement.

1. **Objectifs**

La réalisation de cette application web à plusieurs objectifs :

* Permettre la gestion (ajout, modification et suppression) d’une bibliothèque d’informations à propos de films.
* Mettre en œuvre la persistance des données (C.R.U.D) dans une application web.
* Partir d’une implémentation en programmation fonctionnelle et refactoriser le code en programmation orienté objet.

1. **User stories**

**En tant que [utilisateur anonyme],** je souhaite [créer un compte] afin d’[avoir accès aux fonctionnalités du site].

**En tant que [utilisateur anonyme],** je souhaite [me connecter à mon compte] afin d’[avoir accès aux fonctionnalités du site].

**En tant que [utilisateur connecté],** je souhaite [voir tous les films enregistrés en base de données] afin de [sélectionner un film dans cette liste].

**En tant que [utilisateur connecté],** je souhaite [ajouter un film dans la liste des films] afin de [le retrouver plus tard].

**En tant que [utilisateur connecté],** je souhaite [modifier les informations d’un film] afin de [rectifier une erreur].

**En tant que [utilisateur connecté],** je souhaite [supprimer un film de la liste des films] afin de [ne plus le voir apparaitre dans la liste des films].

**En tant que [utilisateur connecté],** je souhaite [voir les informations de mon compte utilisateur] afin de [vérifier leurs exactitudes].

**En tant que [utilisateur connecté],** je souhaite [modifier les informations de mon compte] afin de [rectifier une erreur ou changer d’avis].

**En tant que [utilisateur connecté],** je souhaite [me déconnecter] afin de [verrouiller les fonctionnalités du site].

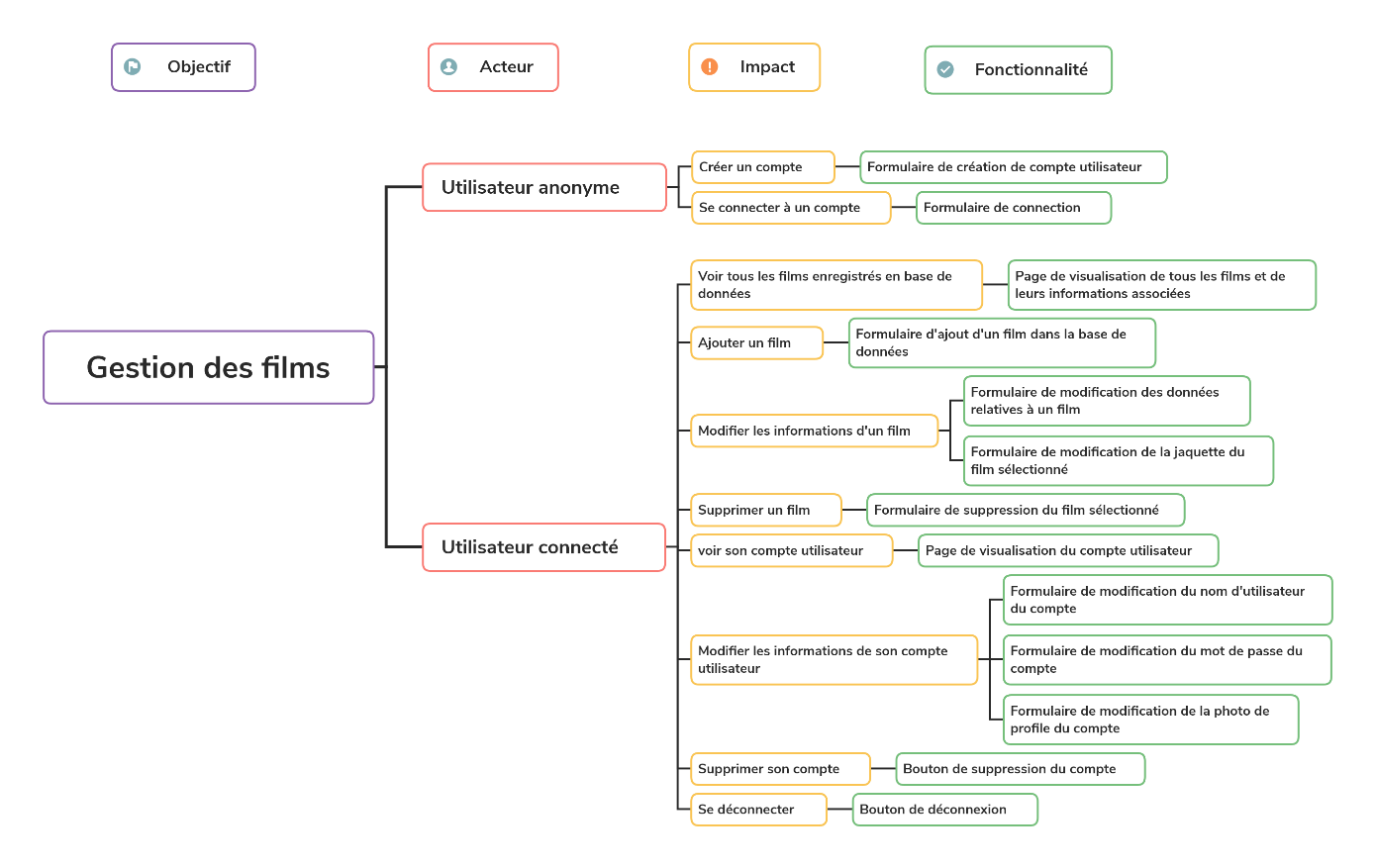
**En tant que [utilisateur connecté],** je souhaite [supprimer mon compte] afin de [ne plus revenir sur ce site].

1. **Impact mapping du projet**

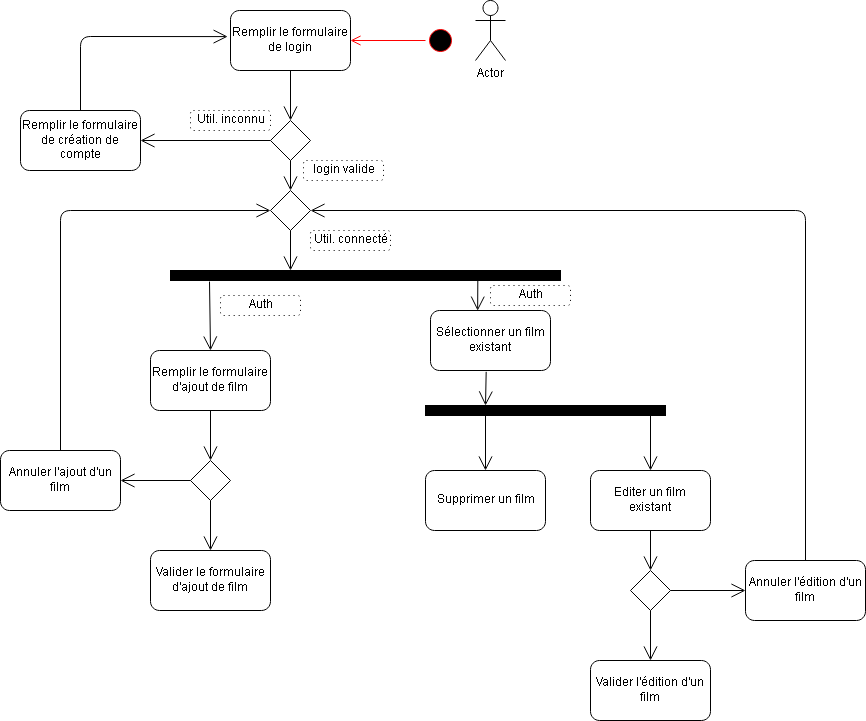
Pour ce projet, j’utilise l’impact mapping (bien qu’orienté business) plutôt qu’un diagramme UML. L’impact mapping à l’avantage d’avoir une mise en œuvre simple et d’être très clair sur le périmètre fonctionnel d’un projet.

L’impact mapping est une technique permettant de représenter sous forme d’un graphique les fondamentaux d’un projet. Il a pour but de se concentrer sur l’essentiel (fonctionnalités) pour ne pas se disperser et répond à 4 grandes questions :

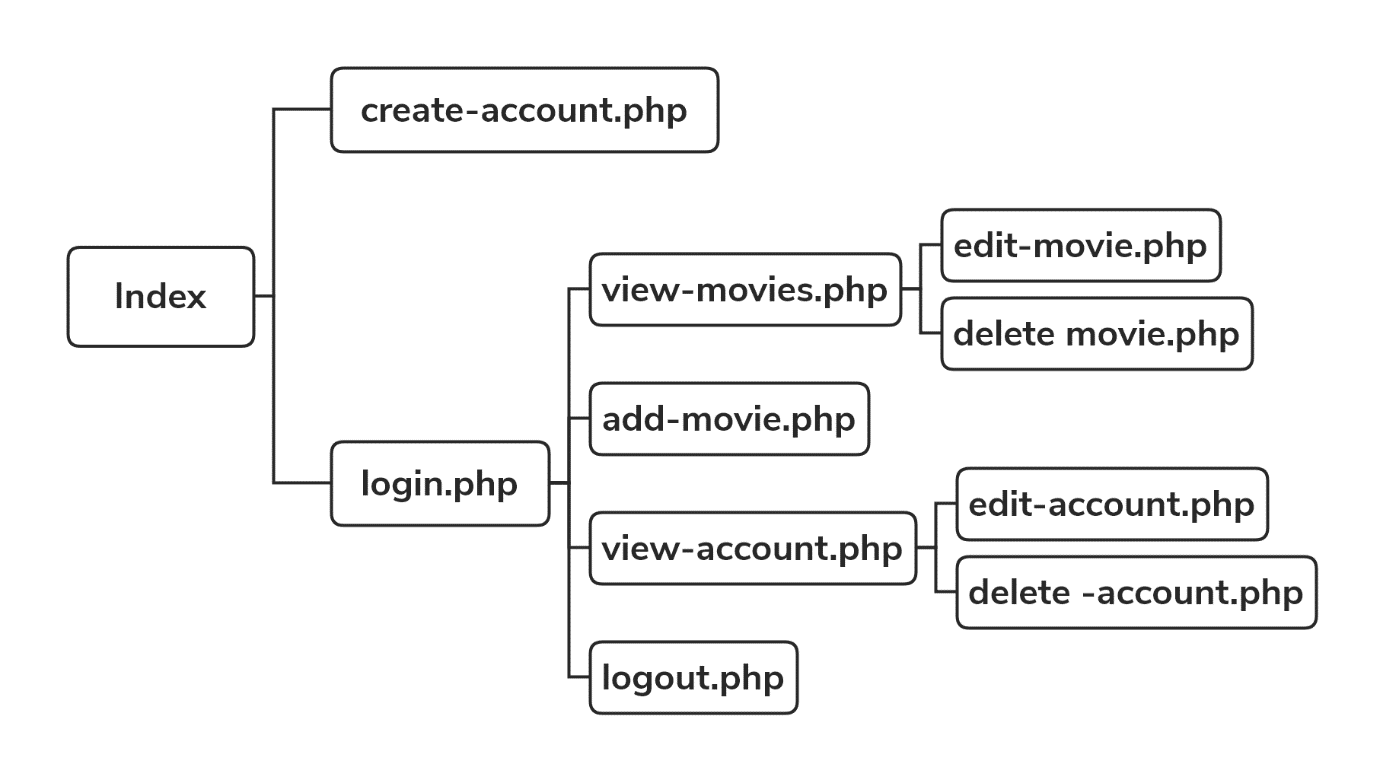
* Pourquoi ? *\*Objectif.* Le fondement même du projet. Le(s) besoin(s) auquel(s) il répond.
* Qui ? *\*Acteur.* Les acteurs, utilisateurs, personnes impactées par le projet.
* Comment ? *\*Impact.* Les comportements qui vont permettre d’atteindre l’objectif.
* Avec quelle(s) fonctionnalité(s) ? *\*Fonctionnalité.* Le(s) fonctionnalité(s) permettant de créer le comportement pour l’impact souhaité.



1. **Diagramme d’activité**

****

1. **Structure des pages**



1. **Responsive**

L’application devra s’adapter aux appareils nomades pour une navigation optimale.

La barre de navigation devra s’adapter en fonction de la largeur et du type d’appareil.

Une attention particulière sera faite à l’adaptation sur mobile.

1. **Accessibilité**

Bien que l’application n’ait pas vocation à être disponible sur internet le respect de quelques recommandations au sujet de l’accessibilité est requis :

* Proposer des équivalents textuels à tout contenu non textuel.
* Rendre la navigation au clavier possible dans toute l’application.
* Ne pas utiliser de contenu susceptible de provoquer des crises convulsives.
* Rendre les contenus textuels lisibles et compréhensibles en utilisant une police sans empattement.

1. **Contraintes des principales fonctionnalités**

|  |  |
| --- | --- |
| **Fonctionnalité** | **Contrainte(s) associée(s)** |
| **Créer un compte utilisateur** | Pas de doublons de noms d’utilisateurs + vérifier les données en entrée. |
| **Se connecter à un compte utilisateur** | Impossible si utilisateur connecté. |
| **Voir la liste de tous les films** | Utilisateur connecté. |
| **Ajouter un film** | Vérifier les données en entrée. |
| **Modifier les informations d’un film** | Vérifier les données en entrée + renommer la jaquette associée. |
| **Modifier la jaquette d’un film** | Extensions autorisées uniquement. |
| **Supprimer un film** | Supprimer la jaquette associée. |
| **Voir les informations de son compte** | Session active. |
| **Modifier le nom d’utilisateur** | Session active + vérifier le mot de passe. |
| **Modifier le mot de passe** | Session active + vérifier les données en entrée + vérifier le mot de passe actuel. |
| **Modifier la photo de profile** | Extensions autorisées uniquement + renommer image. |
| **Se déconnecter** | Suppression de toutes les sessions. |

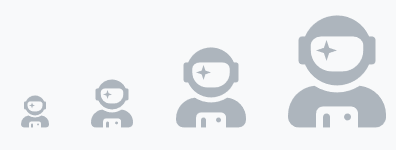
1. **Langue**

La langue de l’application est l’anglais.

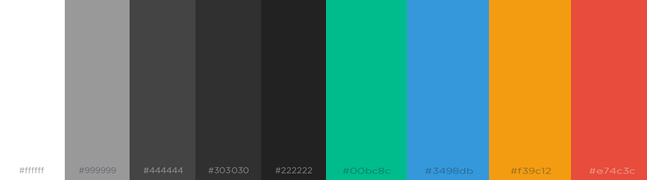
1. **Charte graphique**

**Nom :** Movies Space

**Logo :**



**Palette de couleurs :**

****

**Typographie :**

Toute l’application utilise la police Montserrat

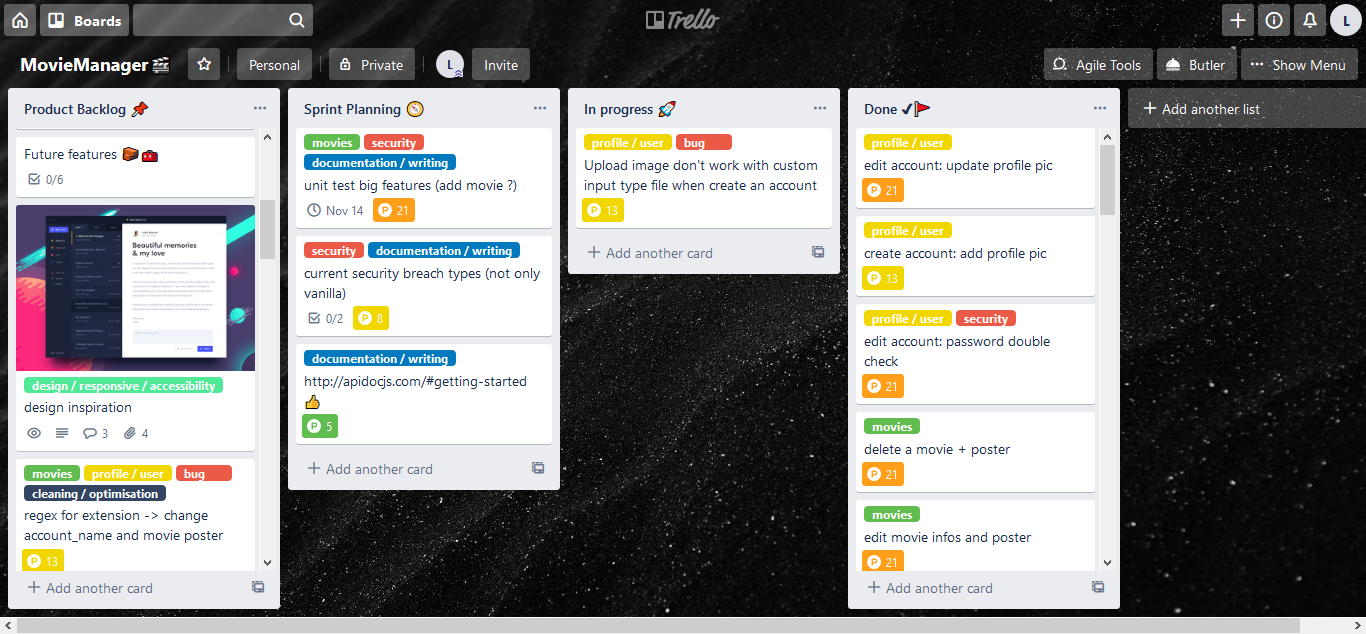
****

## Spécifications techniques du projet

1. **Gestion de projet et organisation**

Pour la gestion du projet j’ai choisi d’utiliser le logiciel Trello, la méthodologie Kanban, la démarche Agile et le Framework Scrum (user story, backlog, sprint).

Organisation du tableau de bord Trello après découpage et estimation des tâches via des « stories points » :



Les colonnes vont permettre d’organiser les tâches concrètes après découpage. J’ai opté pour une organisation en 4 colonnes :

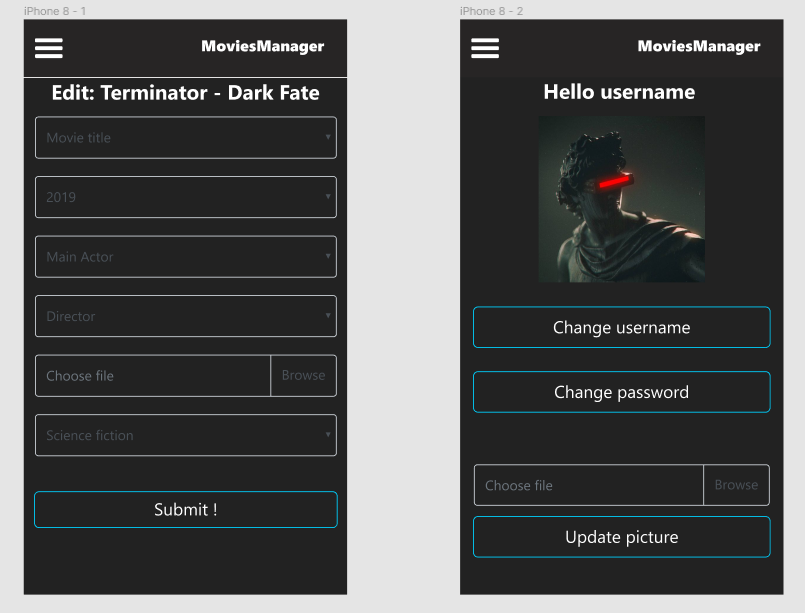
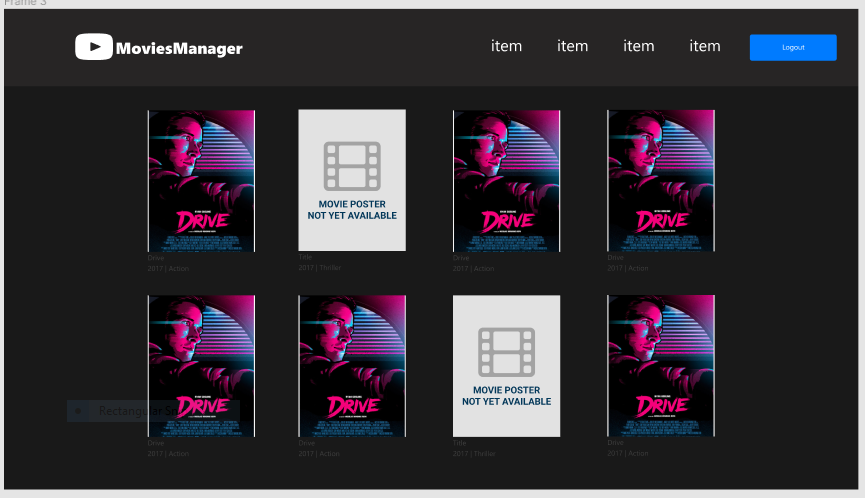
* Product Backlog : ce que je dois implémenter, découper en très petites tâches. Cette liste comprend aussi mes idées, listes, tâches, pense-bête en vrac.
* Sprint Planning : Les tâches du sprint actuel (j’ai décidé qu’un sprint est égal à une semaine pour ce projet)
* In Progress : ce que je fais en ce moment. Pas plus de 2 cartes du sprint en cours en même temps dans cette colonne.
* Done : les tâches faites sont transférées dans cette colonne.

Les tags permettent de regrouper les cartes par thème / domaine.

*“Scrum, c’est comme le poker, vous apprenez les règles en 10 minutes, mais vous mettez du temps pour devenir vraiment bon. David Matthew ».*

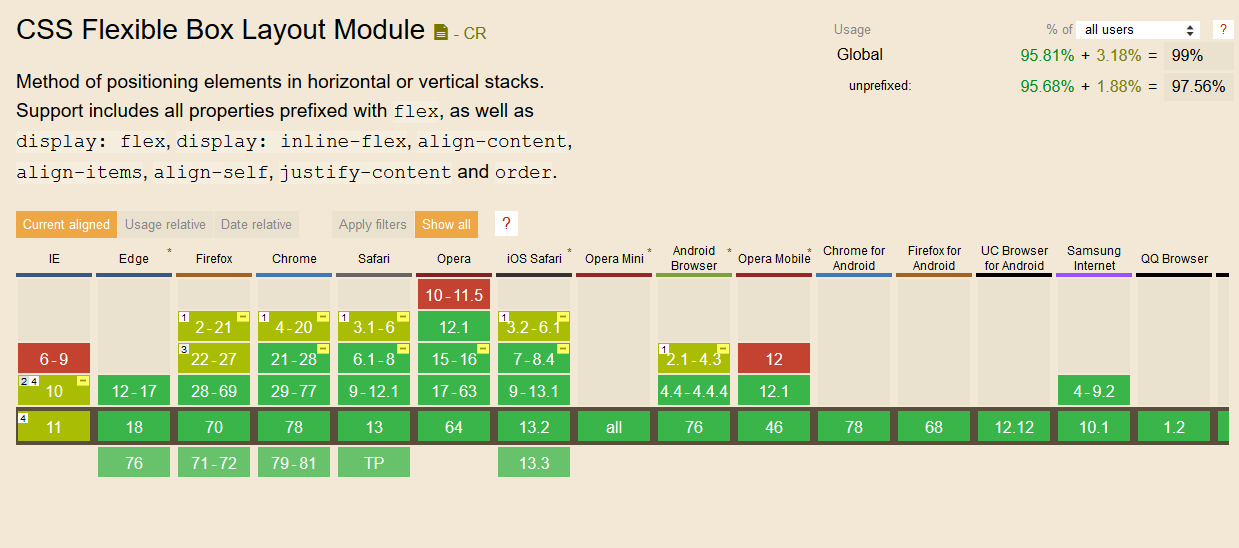
1. **Maquettage**

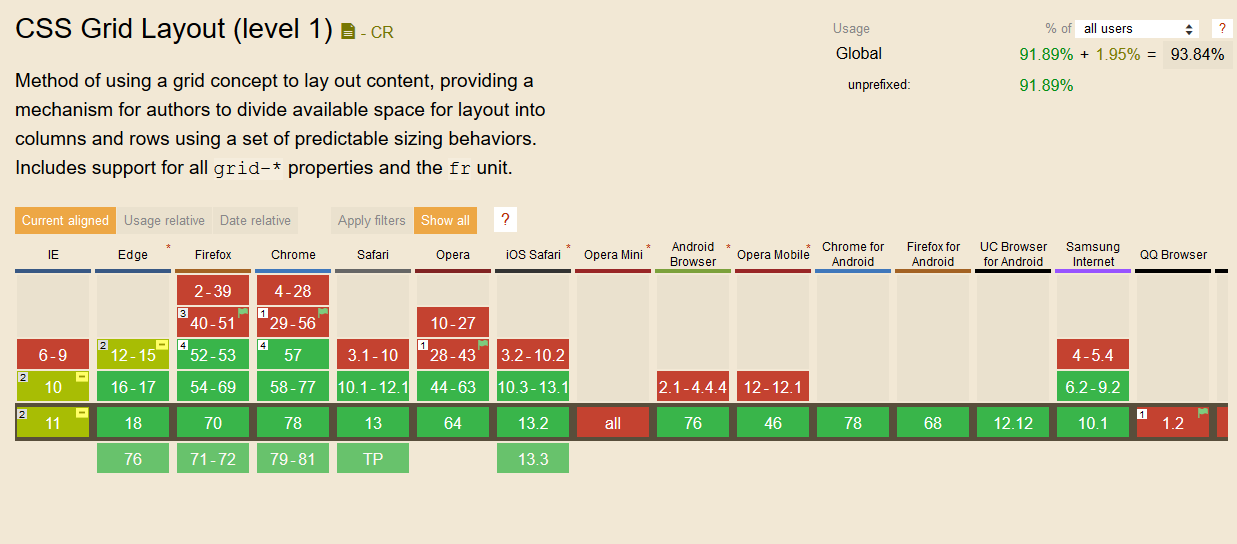
Le maquettage est réalisé avec le logiciel Figma ainsi que la librairie de composants Bootstrap de Figma.

****

1. **Supports / navigateurs**

L’interface et les fonctionnalités devront être compatible avec les navigateurs récents supportant CSS Flexbox et Grid :



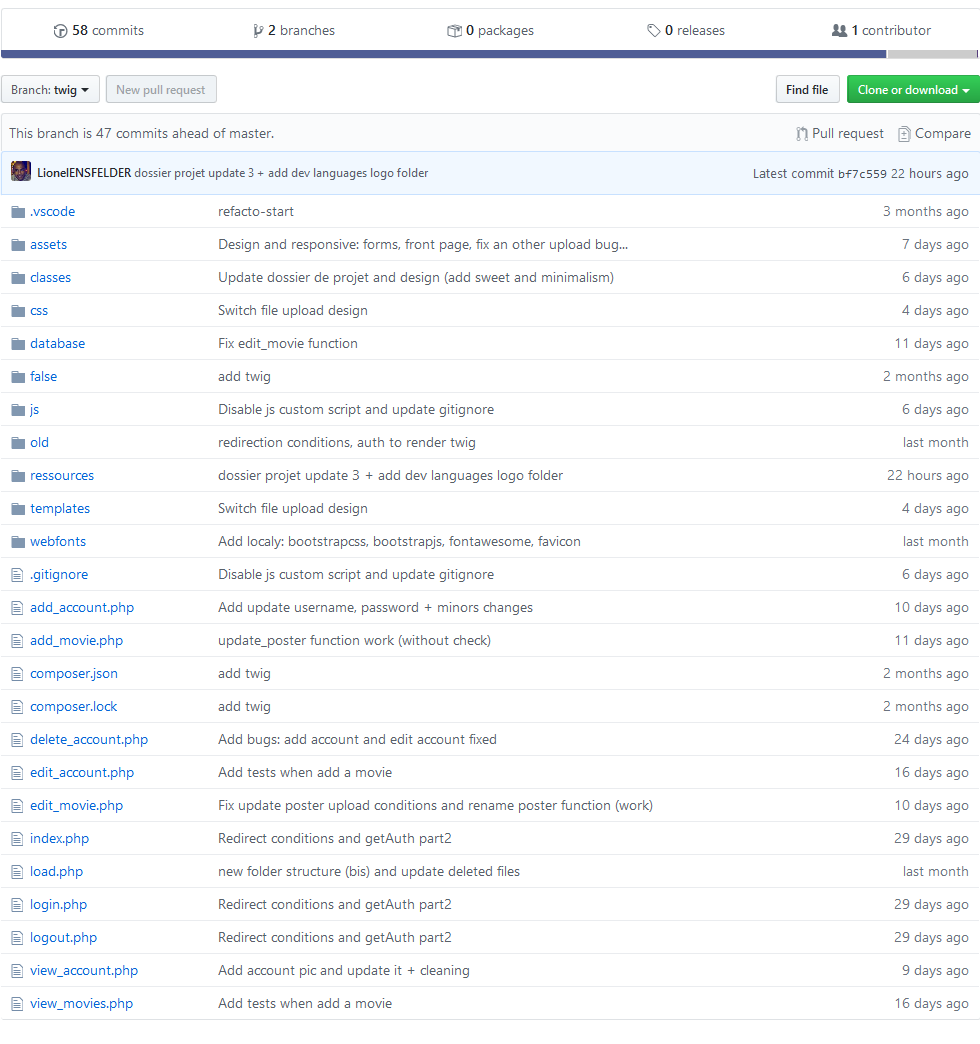


1. **Versionning**

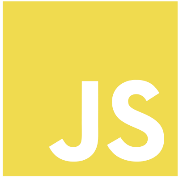
Le versionning du projet est assuré par git et Github.

La branche master sera la branche principale durant le développement de la première version qui sera implémentée en programmation fonctionnelle.

Au fur et à mesure des refactorisations, une branche sera créée pour chaque changement majeur. Une intégration du moteur de templates Twig fera également l’objet d’une branche distincte.

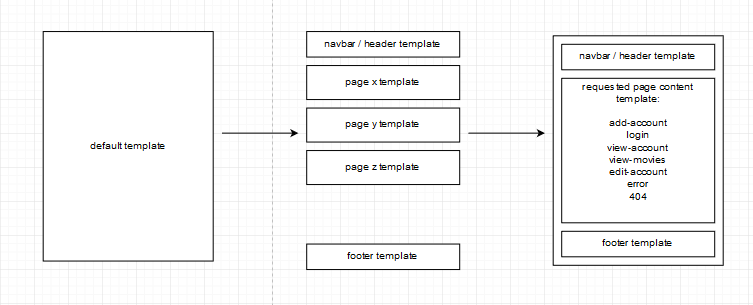


1. **Langages de programmation utilisés**

****

**Boostwatch :** afin de coller au design sur fond noir souhaité sans augmenter le temps d’implémentation de l’interface, un thème fournit par Bootswatch ([Darkly theme](https://bootswatch.com/darkly/)) sera utilisé. Celui-ci est spécialement conçu pour surcharger le CSS de base du Framework Bootstrap.

**Twig :** Twig est un moteur de templates pour le langage de programmation PHP utilisé par défaut par le Framework Symfony. Il permettra de créer des composants graphiques réutilisables et flexibles. La fonctionnalité d’héritage de template servira ici à imbriquer des composants (qui sont aussi des templates) tel qu’une barre de navigation ou un pied de page à un modèle général qui sera la base de toutes les pages.



1. **Serveur Web**

**uWamp :** uWamp est un serveur Wamp Apache MySQL PHP et SQLite. Il est destiné à être lancer depuis une clé USB ce qui le rend facilement portable. Cette application est parfaitement adaptée à ce projet car le temps de travail sera partagé entre le centre AFPA, le domicile et, éventuellement, le lieu du stage si besoin.

**Structure des dossiers du serveur :**

* bin : serveur Web, les bases de données et les versions de PHP
* phpapps : applications préinstallé php (ici phpmyadmin).
* utils : contient des outils en plus.
* www : répertoire contenant le projet.
* UwAmp : interface de control du serveur + executable.

**Configuration du serveur :**

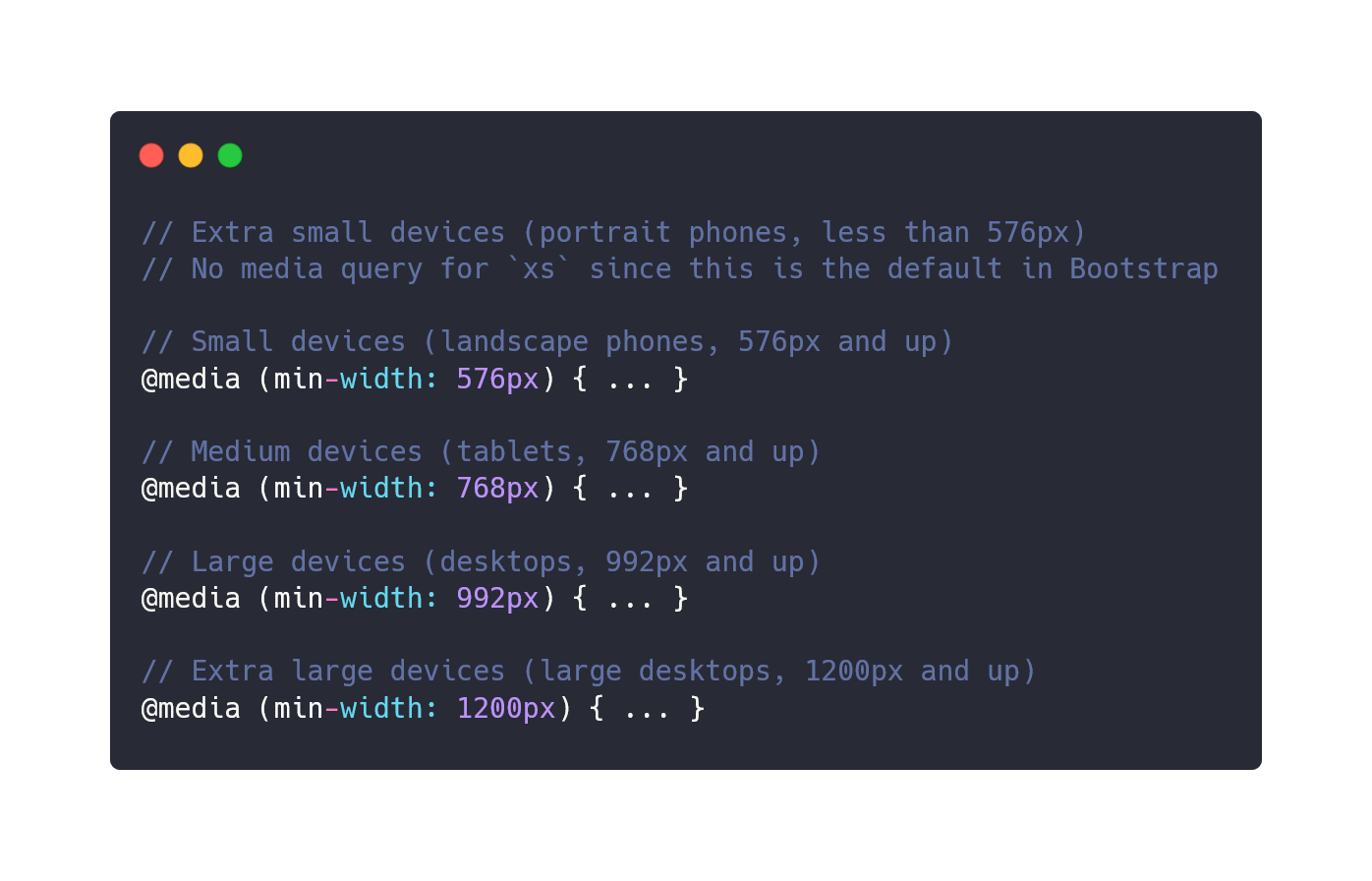
* Fichier de configuration apache : bin/apache/conf/httpd\_uwamp.conf
* Fichier de configuration PHP : bin/php/php\_[\*]/php\_uwamp.ini
* Fichier de configuration MySQL : database/mysql-\*/my\_uwamp.ini

**Versions :**

* PHP 7.2.7
* MySQL 5.7.11

1. **Responsive / web mobile**

Pour assurer l’adaptation de l’interface de l’application aux différentes tailles d’écran nous utiliserons les points de ruptures (responsive breackpoints) fournis par le Framework Bootstrap.



1. **Sécurité**

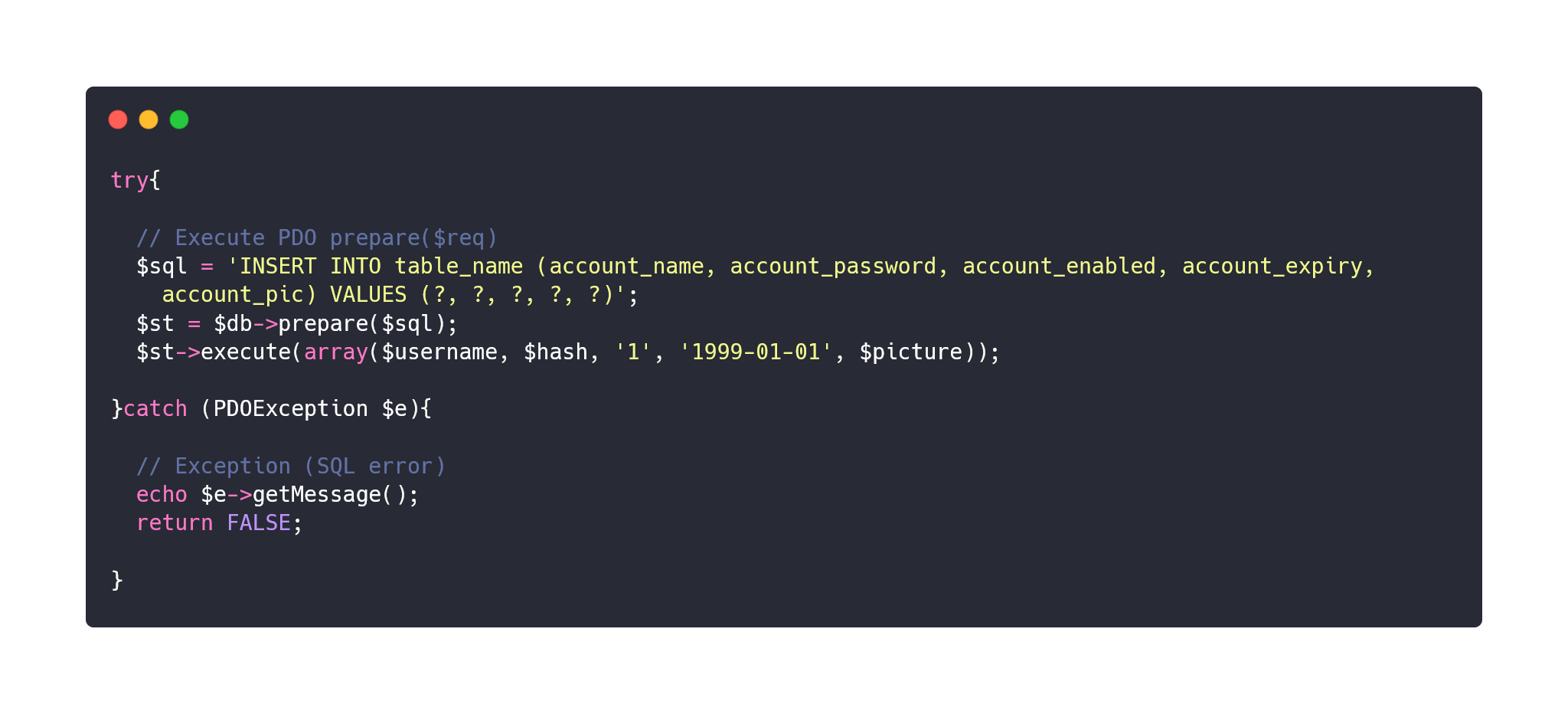
**Filtres de nettoyage PHP**

PHP permet de filtrer les données, soit en les validant, soit en les nettoyant. Les données issues des formulaires de l’application seront nettoyées en utilisant FILTER\_SANITIZE().

****

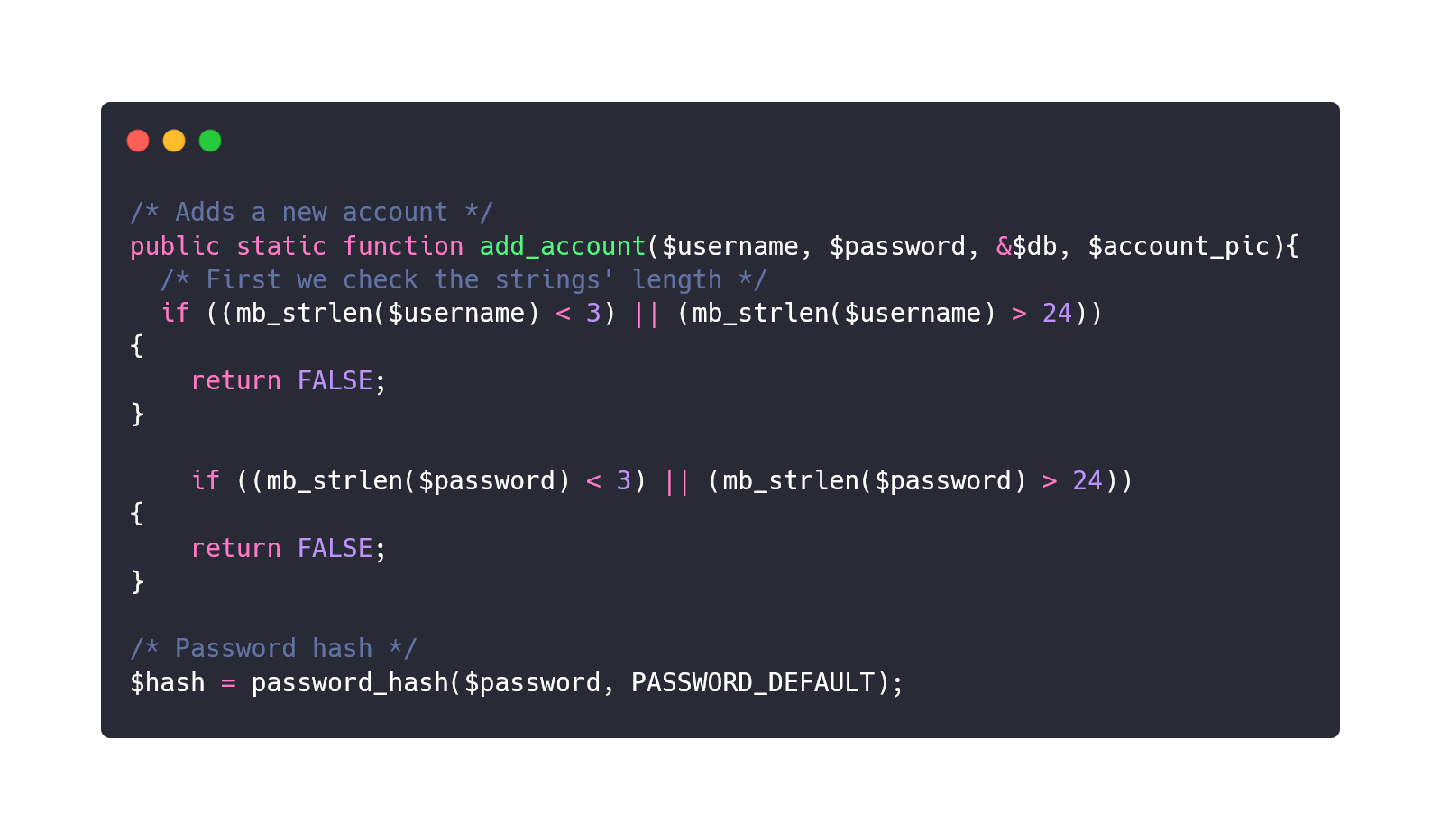
**Requêtes préparées (PDO)**

Les requêtes préparées sont des requêtes stockées en mémoire que l’on pourra exécuter plusieurs fois. Une requête préparée permet en plus de ne plus traiter les données comme faisant partie de la requête mais vraiment comme des données. Lors de la préparation d’une requête la base de données crée en réalité des cases dans lesquelles elle va « binder » les valeurs qu’on lui donne. Il n’y a donc plus moyen de pirater une requête avec des valeurs frauduleuses. Ces requêtes sont analysées (préparée) une seule fois et pourrons être appelées plus tard. Enfin ce type requêtes utilisent moins de ressources et s’exécute plus rapidement.



**Mots de passe**

Lors de l’ajout d’un compte utilisateur, le mot de passe de celui-ci sera traité de façon à être chiffré. Ce chiffrement sera assuré par une fonction de hachage fournit nativement par PHP : password\_hash().L’algorithme utiliser sera celui actif par défaut (PASSWORD\_DEFAULT) pour cette fonction. Il est conçu pour changer dans le temps selon les algorithmes ajoutés à PHP. Pour cette raison il sera donc stocké dans la base de données dans une colonne pouvant contenir 255 caractères.

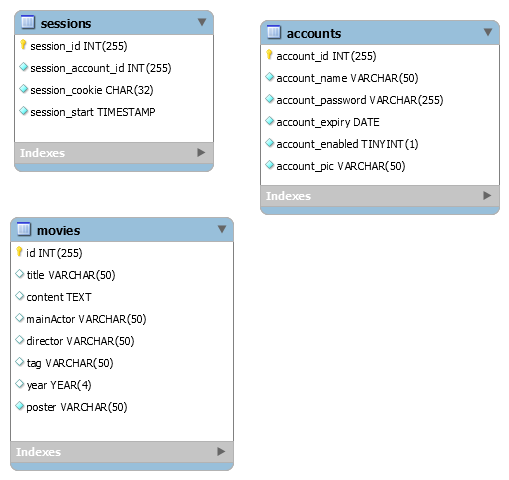


1. **Accessibilité**

* Menu toujours disponible en haut.
* Toutes les images affichées dans l’interface seront implémentées de façon à faciliter la navigation aux personnes ayant un handicap. Elles seront munies de balises [alt] et [title].
* Navigation au clavier (tabindex) suivant l’ordre naturel des éléments de chaque page de gauche à droite (barre de navigation, éléments de la page, fenêtres modales, pieds de pages). Sauf besoin particulier l’ajout d’attributs ‘’tabindex’’ supplémentaire ne sera pas nécessaire.



1. **Organisation des données et base de données**



**Schémas physiques des données**

* Table movies

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Code** | **Désignation** | **Type** | **Taille** | **Remarque / défaut** |
| id | Identifiant unique du film | INT | 255 | AUTO\_INCREMENT |
| title | Titre du film | VARCHAR | 50 |  |
| content | Résumé du film | TEXT | - |  |
| mainActor | Acteur | VARCHAR | 50 |  |
| director | Réalisateur | VARCHAR | 50 |  |
| tag | Tag | VARCHAR | 50 |  |
| year | Année | YEAR | 4 |  |
| poster | URL de la jaquette | VARCHAR | 50 | assets/posters/default.jpg   |  |  | | --- | --- | |  |  | |

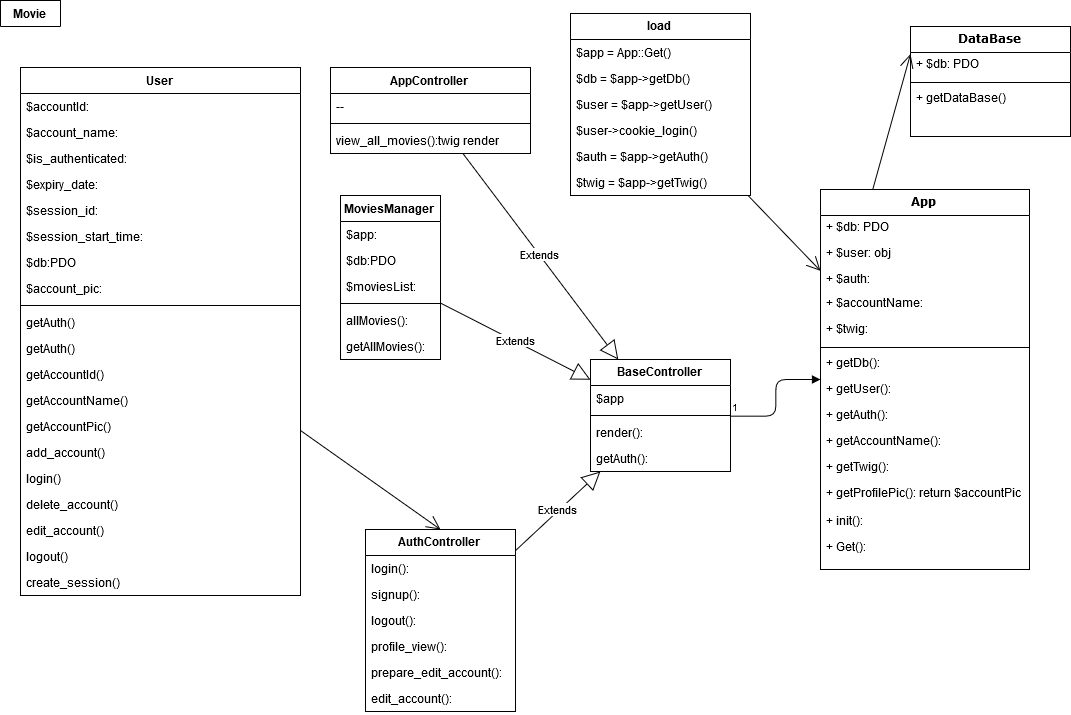
* Table accounts

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Code** | **Désignation** | **Type** | **Taille** | **Remarque / défaut** |
| account\_id | Identifiant | INT | 255 | AUTO\_INCREMENT |
| account\_name | Nom d’utilisateur | VARCHAR | 50 |  |
| account\_password | Mot de passe | VARCHAR | 255 |  |
| account\_expiry | Date d’expiration | DATE | - | 1999-01-01 |
| account\_enabled | Activé / désactivé | TINYINT | 1 | 0 |
| account\_pic | Photo de profile | VARCHAR | 50 |  |

* Table sessions

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Code** | **Désignation** | **Type** | **Taille** | **Remarque / défaut** |
| session\_id | Identifiant de session | INT | 255 | AUTO\_INCREMENT |
| session\_account\_id | Identifiant du compte | INT | 255 |  |
| session\_cookie | Cookie de session | CHAR | 32 |  |
| session\_start | Date et heure de début de session | TIMESTAMP | - | CURRENT\_TIMESTAMP |

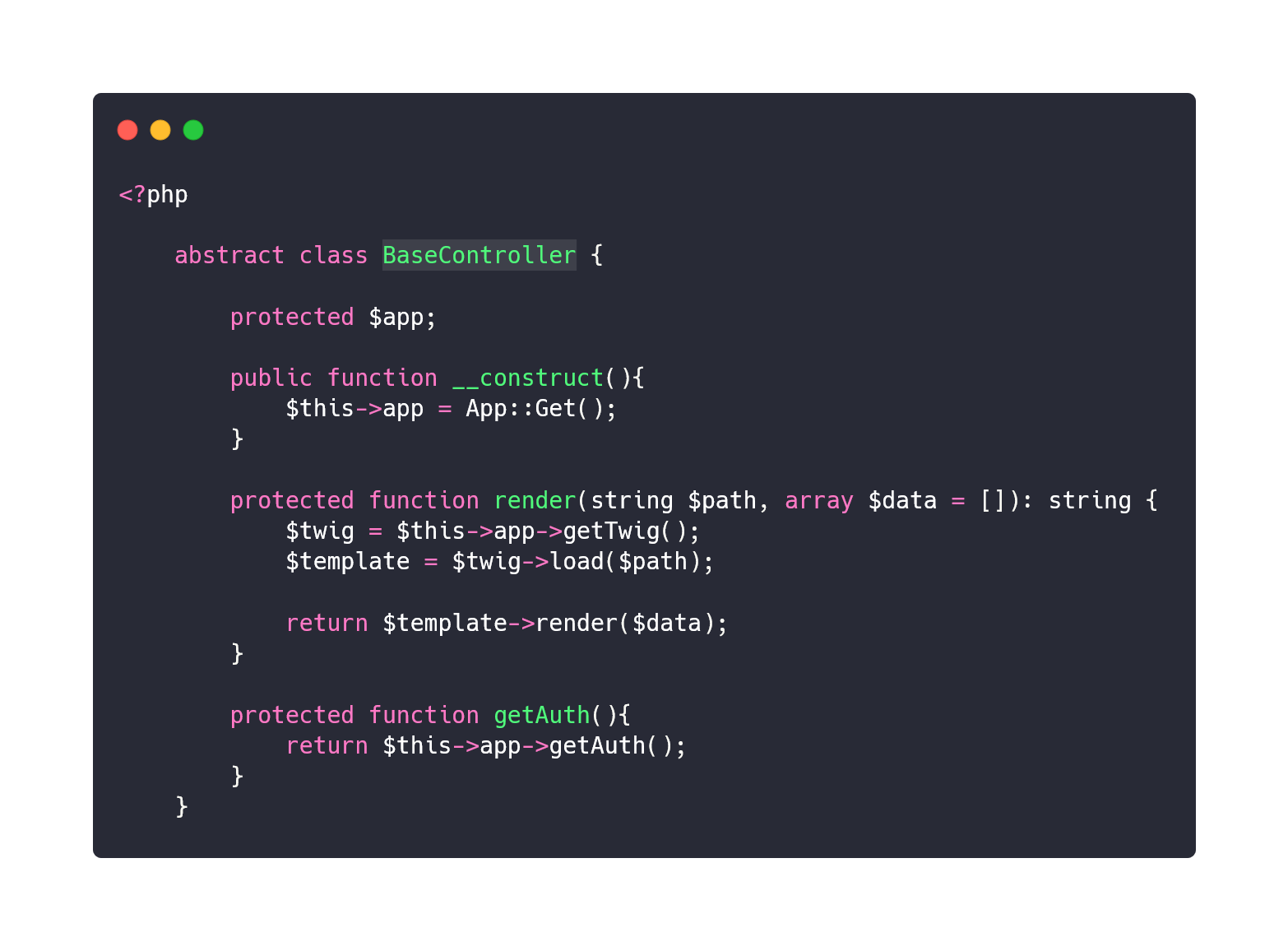
**Diagramme de classes**



# Chapitre 2 – Présentation des éléments de réalisation

## Fonctionnement globale de l’application

* **BaseController** est le contrôleur de base duquel héritent les contrôleurs réels de l’application. Il centralise les fonctionnalités partagées par tous les contrôleurs.

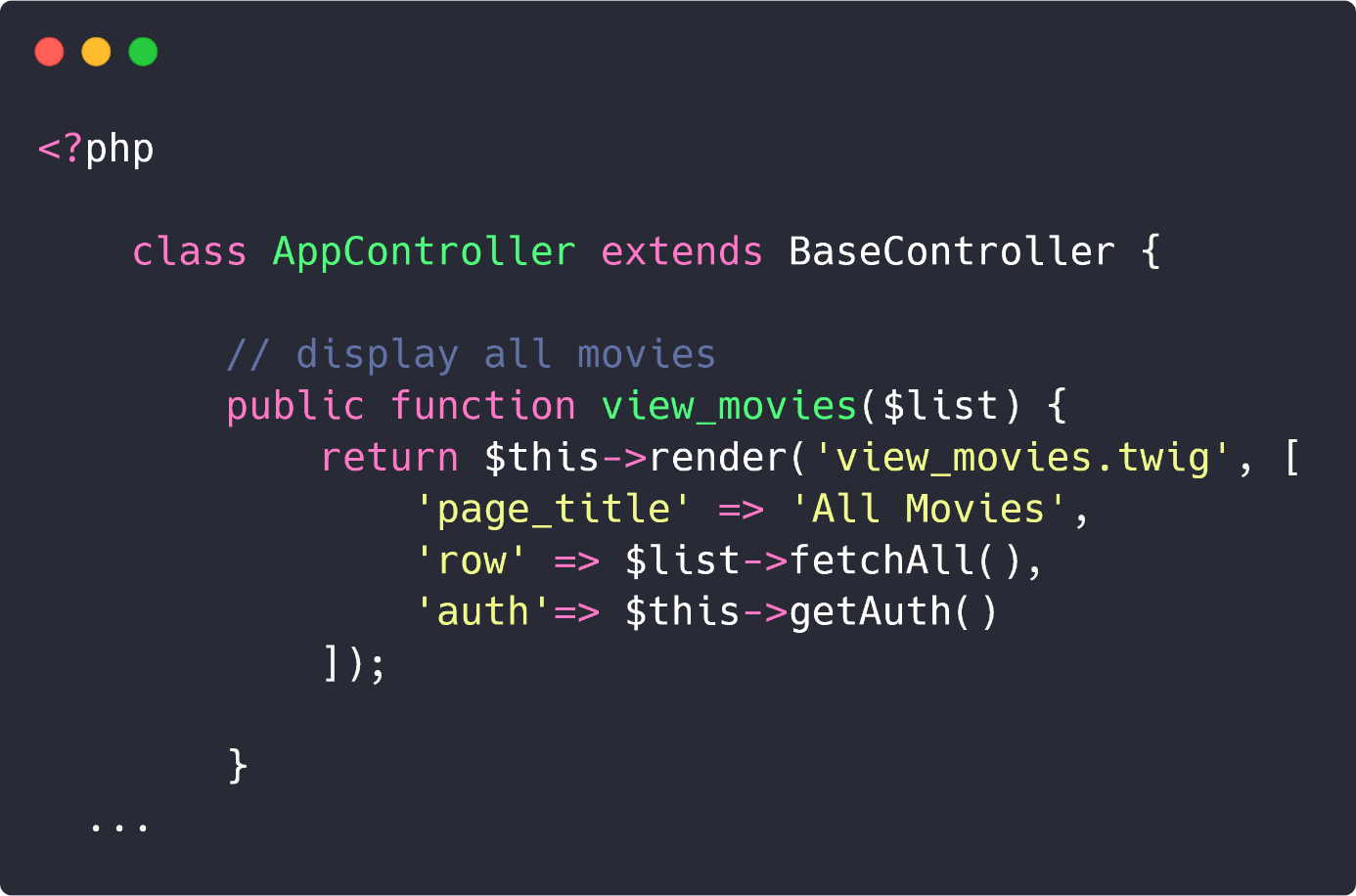


1. **Front-end**

* **Twig** est la classe du moteur de templates.
* **AppController** gère le rendu graphique des pages relatives aux films.
* **AuthController** gère les opérations relatives à un utilisateur et au rendu des pages relatives au l’utilisateur.

Ci-dessous un exemple de la page *view\_movies.php* qui renvoie vers la page de login si l’utilisateur n’est pas connecté, ou, se charge de récupérer la liste des films et de l’affiché si celui-ci est connecté.





1. **Back-end**

* **DataBase** est la classe qui renvoie une instance de PDO sous la forme d’un Singleton.
* **App** est le conteneur de l'ensemble des constantes, services, helpers utiles à travers l'application.
* **Load** est un « autoload » des classes et chargement de 'vendor/autoload.php' pour le moteur de templates Twig.
* **MoviesManage**r gère les opérations en base de données concernant les films.
* **User** est une entité qui représente un utilisateur.

Ci-dessous une partie du conteneur App avec des exemples de getters permettant de d’accéder à certaines variables dans toute l’application.



Ici un exemple de la fonction de suppression d’un film et de son poster associé.



Ci-dessous un exemple de la création d’une requête SQL pour l’édition d’un compte en fonction des arguments passés.

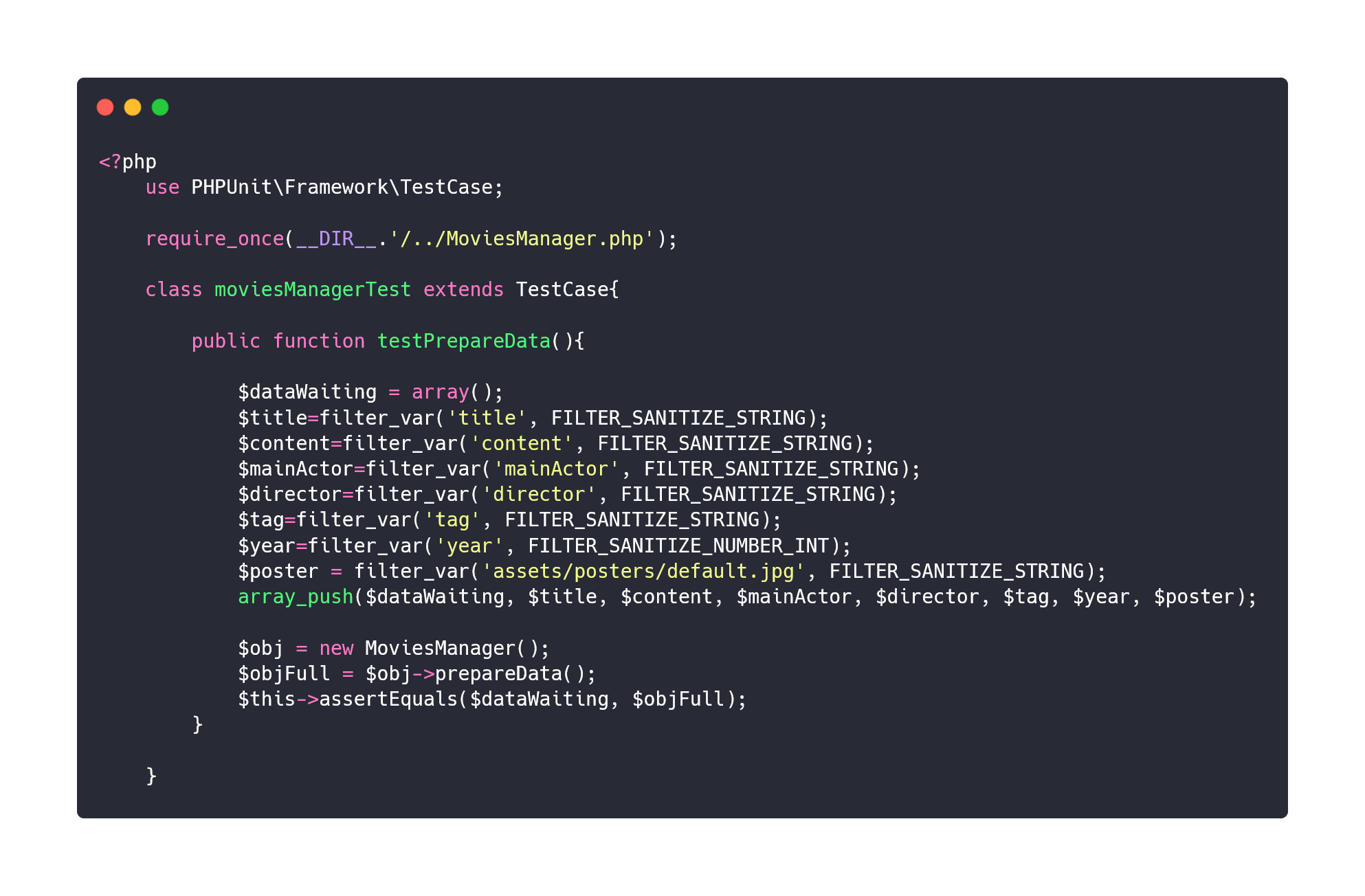


## Tests unitaires

L’une des principales fonctionnalités de mon application est l’ajout d’un film dans la base de données.

Lors de l’implémentation des tests, j’ai isolé la fonction qui prépare les données provenant du formulaire d’ajout de films puis retourne un tableau. Ainsi j’ai pu comparer le tableau des valeurs en entrée avec celui des valeurs en sortie.

Ci-dessous la classe qui effectue le test.



1. **Tester les filtres de nettoyage PHP en ajoutant des balises HTML.**





1. **Comparer les valeurs d’entrée et de retour.**





# Chapitre 3 – Description et mise en œuvre des compétences transversales

1. Description de la veille sur les vulnérabilités de sécurité liée au développement d’une interface utilisateur web dynamique

Pour effectuer une veille technologique et plus particulièrement, une veille sur les vulnérabilités liées au développement d’une interface utilisateur web dynamique, j’utilise en premier lieu un agrégateur de flux.

J’utilise Feedly parce qu’il est en ligne, donc accessible partout. Il dispose d’une fonctionnalité de recherche de sujets intégré faisant remonter les sources (flux rss par site) correspondantes et la possibilité de séparer les flux d’information dans des collections bien définies.

J’utilise aussi le site stackoverflow.com pour des recherches précises, poser des questions par rapport à un problème que je rencontre et le suivi de topics (sujets, mots clé, langage de programmation, technologie etc…).

Lors de ma veille j’ai identifié les failles de sécurité les plus courantes :

* **CSRF**, Cross Site Request Forgery (falsification de requête intersite). Permet à un attaquant de forcer ses victimes à effectuer certaines actions sur un site cible, sans qu’elles ne s’en rendent compte.
* **XSS**, Cross Site Scripting.

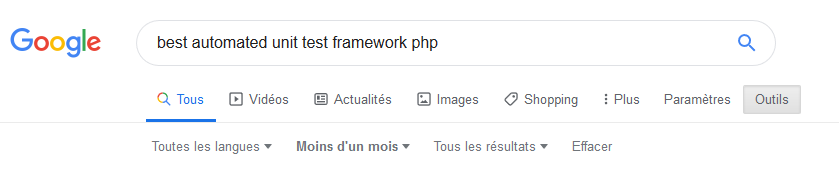
Il existe 3 sous-types d’attaques XSS : attaques XSS stockées (stored XSS attacks), attaques XSS reflétées (reflected XSS attacks), attaques XSS basées sur le DOM (DOM based XSS).

* **Injections** : le but est d’insérer des données en entrée d’une fonction, d’un programme ou bien d’un script afin de les détourner de leur fonction d’origine.
* **Hijacking** : il s’agit de la possibilité pour un pirate d**’utiliser l’identité d’autres personnes sur un site Internet**
* [**Broken Authentification and Session Management**](https://www.httpcs.com/fr/faille-cross-site-scripting-xss)correspond au risque de casser ou de contourner la gestion de l’authentification et de la session. Comprend notamment le vol de session ou la récupération de mots de passe.
* **Sensitive Data Exposure** fait référence aux failles de sécurité exposant des données sensibles comme les mots de passe, les numéros de carte de paiement ou encore les données personnelles et la nécessité de chiffrer ces données.

En ce qui concerne les recommandations de sécurité je m’appuis aussi sur le guide OWASP.

## Description d’une situation ayant nécessité une recherche à partir d’un site anglophone

Lors du développement de l’application, j’ai eu besoin de faire des tests unitaires pour vérifier le bon fonctionnement des principales classes et méthodes. J’ai d’abord fait des recherches sur la meilleure façon de structurer une recherche sur Google, puis sur les tests en générale dans le développement web et enfin plus particulièrement sur les tests unitaires en PHP.Pour effectuer ma recherche sur les tests unitaire j’ai utilisé l’expression « *best automated unit test framework php*». J’ai ensuite trié les résultats par date sans distinction de langue.



1. Extrait du site anglophone utilisé pour la recherche décrite précédemment.

**Ci-dessous un extrait de la documentation du site PHPUnit.**

*This appendix lists the various assertion methods that are available.*

*Static vs. Non-Static Usage of Assertion Methods.*

*PHPUnit’s assertions are implemented in PHPUnit\Framework\Assert. PHPUnit\Framework\TestCase inherits from PHPUnit\Framework\Assert.*

*The assertion methods are declared static and can be invoked from any context using PHPUnit\Framework\Assert::assertTrue(), for instance, or using $this->assertTrue() or self::assertTrue(), for instance, in a class that extends PHPUnit\Framework\TestCase.*

*In fact, you can even use global function wrappers such as assertTrue() in any context (including classes that extend PHPUnit\Framework\TestCase) when you (manually) include the src/Framework/Assert/Functions.php sourcecode file that comes with PHPUnit*

**Ci-dessous la traduction de cet extrait.**

*« Cet appendice liste les différentes assertions possibles.*

*Utilisation des assertions statiques contre les assertions non statiques.*

*Les assertions de PHPUnit sont implémentées dans le fichier PHPUnit\Framework\Asser. PHPUnit\Framework\TestCase hérite de PHPUnit\Framework\Assert.*

*Les méthodes d'assertion sont déclarées de façon statique et peuvent être instanciées depuis n'importe quel contexte en utilisant PHPUnit\Framework\Assert::assertTrue(), pour l'instance, ou en utilisant $this->assertTrue() ou encore self::assertTrue(), pour une instance dans une classe qui étend PHPUnit\Framework\TestCase.*

*En effet, vous pouvez utiliser les wrappers de fonction globales tels que assertTrue() dans n'importe quel contexte (incluant les classes qui étendent PHPUnit\Framework\TestCase) quand vous incluez (manuellement) le code source du fichier src/Framework/Assert/Functions.php livré avec PHPUnit. ».*