Rapport de Stage

**2012**

Pierre EVERS et Maxime DESMAUTS

L3 informatique

31/05/2012



# Remerciements

Nous tenons tous d’abord à remercier Monsieur Adrien Goëffon pour le temps qu’il nous a consacré afin de mener à bien notre stage ainsi qu’à l’université d’Angers et évidemment tout le personnel enseignant nous ayant apporté les connaissances.

Sommaire

[Remerciements 2](#_Toc326158835)

[1-Introduction 5](#_Toc326158836)

[2-Présentation du Projet 6](#_Toc326158837)

[2.1) Rappel du contexte 6](#_Toc326158838)

[2.2) Présentation du sujet 6](#_Toc326158839)

[2.3) Les besoins 6](#_Toc326158840)

[2.4) Langages utilisés 7](#_Toc326158841)

[3-Réalisation du Projet 8](#_Toc326158842)

[3.1) Les contraintes techniques 8](#_Toc326158843)

[3.2) Conception de la plateforme 8](#_Toc326158844)

[3.3) Réalisation de la base de donnée 9](#_Toc326158845)

[3.5) Mise en place de la partie Film 9](#_Toc326158846)

[3.6) Mise en place de l’inscription 11](#_Toc326158847)

[3.6) Mise en place de la gestion des utilisateurs 11](#_Toc326158848)

[3.7) Mise en place de l’ajout de film 12](#_Toc326158849)

[3.8) Mise en place de la liste de favoris 14](#_Toc326158850)

[3.9) Mise en place de la notation d’un film 15](#_Toc326158851)

[3.10) Mise en place des statistiques 16](#_Toc326158852)

[3.11) CSS et design du site 17](#_Toc326158853)

[3.12) Gestion des tables 17](#_Toc326158854)

[4-Tests effectués 18](#_Toc326158855)

[5-Evolutions possible de la plateforme 19](#_Toc326158856)

[6-Conclusion 20](#_Toc326158857)

[7- Bibliographie 21](#_Toc326158858)

[8- Annexes 22](#_Toc326158859)

# 1-Introduction

Durant la période du 23 Avril 2012 au 31 Mai 2012, nous avons eu la chance de réaliser un stage au sein de l’université d’Angers afin de valider notre dernière année de licence Informatique.

Au cours de cette période, notre tâche était de réaliser une application web collaborative permettant l’ajout, la modification et l’affichage des données de films. Cette application devant être évolutive.

Ce document a pour objectif de présenter notre travail effectué et de décrire le déroulement de celui-ci durant toute la période de notre stage.

En vue de rendre compte de manière fidèle et analytique de ces cinq semaines de stage, il nous apparaît logique de commencer par présenter notre sujet de stage et le contexte, avant de nous focaliser sur le développement de notre application et d’en expliquer le déroulement.

# 2-Présentation du Projet

## 2.1) Rappel du contexte

Nous sommes deux étudiants de l’université d’Angers en 3ème année de Licence mention informatique et afin de valider notre diplôme nous avons dû effectuer un stage. Et cn ce qui le concerne, nous avons été encadré par Monsieur Goëffon.

## 2.2) Présentation du sujet

Il s'agit de réaliser une application / site web permettant de gérer de manière collaborative un catalogue de films. En fonction des droits des utilisateurs, ceux-ci pourront ajouter / modifier des données relatives aux films (qui pourront être récupérées automatiquement sur imdb.com par exemple), ou bien saisir une note d'appréciation. Il devra être possible ensuite de générer des listes ou des statistiques à partir de l'ensemble des données. L'application devra être simple d'utilisation et facilement évolutive.

## 2.3) Les besoins

Les objectifs de notre stage sont de concevoir et réaliser une plateforme web collaborative afin de permettre l’ajout, la suppression, la modification de film. On doit pouvoir accéder aux données du film de façon simple. On doit pouvoir rechercher des films en fonction de leur titre. Il doit être possible de noter les films.

Un film possède différentes informations tels que le réalisateur, son année de sortie ou encore ses acteurs, un résumer et les notes attribué par les utilisateurs, il nous faut pouvoir afficher de manière simple ces données.

Il nous est alors demandé de réaliser un système d’authentification avec différents niveaux d’utilisateurs. En fonction de son niveau un utilisateur aura accès à différente partie de la plateforme. Un utilisateur non connecté ne pourra qu’avoir accès aux données des films sans pouvoir les noter. Les utilisateurs connectés eux sont répartie sur 3 niveaux :

-Niveau un, il est possible de noter les films

-Niveau deux, possède les fonctionnalités du niveau un avec en plus la possibilité d’ajouter, et de modifier un film.

-Niveau trois, possède également les fonctionnalités du Niveaux deux mais il peut également gérer les niveaux des autres utilisateurs ainsi que les supprimer, on peut donc le voir comme l’administrateur du site.

Les utilisateurs doivent pouvoir ajouter un film à leurs favoris. Ils doivent ainsi posséder une liste de leurs films favoris.

Il est également demandé de mettre en valeur un aspect communautaire. Afin de réaliser ce besoin il nous est demandé de mettre en place une fonction de groupe. Chaque utilisateur doit donc pouvoir rejoindre un groupe, ou créer un groupe. Une fois dans un groupe il doit pouvoir voir les autres membres du groupe. Ils doivent également pouvoir voir les listes des films favoris des autres membres.

Une fois ces données recueillies sur les utilisateurs et les notations des films, il nous est demandé de générer des statistiques.

Dans un dernier temps, si le temps nous le permet, notre maitre de stage souhaiterait pouvoir modifier les bases de données directement à partir de la plateforme.

Le dernier objectif est que la plateforme soit simple d’utilisation et facilement évolutives.

## 2.4) Langages utilisés

Les langages utilisés pour réaliser cette plateforme sont principalement le PHP, le JavaScript, l’Ajax ainsi que l’utilisation de frameworks tel que dojo pour réaliser les formulaires ou hightcharts pour la partie statistique. Ces frameworks utilisent du javascript : dojo permet de vérifier les données entrées par l’utilisateur dans un formulaire. Highchart génère des diagrammes à partir de données envoyées en norme Json. La partie PHP est codée en orienté objet. Nous avons également utilisé du css 3 pour la réalisation du design de la plateforme.

# 3-Réalisation du Projet

## Les contraintes techniques

Afin de mener à bien notre stage, nous avons dû respecter plusieurs contraintes techniques. Premièrement ils nous étaient demandés de réaliser une application web et donc de savoir maîtriser les langages de programmations web. L’application devant utiliser une large base de données, nous avons donc d’abord dû passer par une phase de conception de la base de données. Comme nous étions deux à travailler sur le projet il a également fallut que l’on mette en place un partage des tâches : pour cela, nous avons opté pour SVN via GoogleCode.

## Conception de la plateforme

Afin de concevoir une plateforme simple d’utilisation et surtout qui soit facilement modifiable, nous avons adopté plusieurs normes.

-Modèle MVC

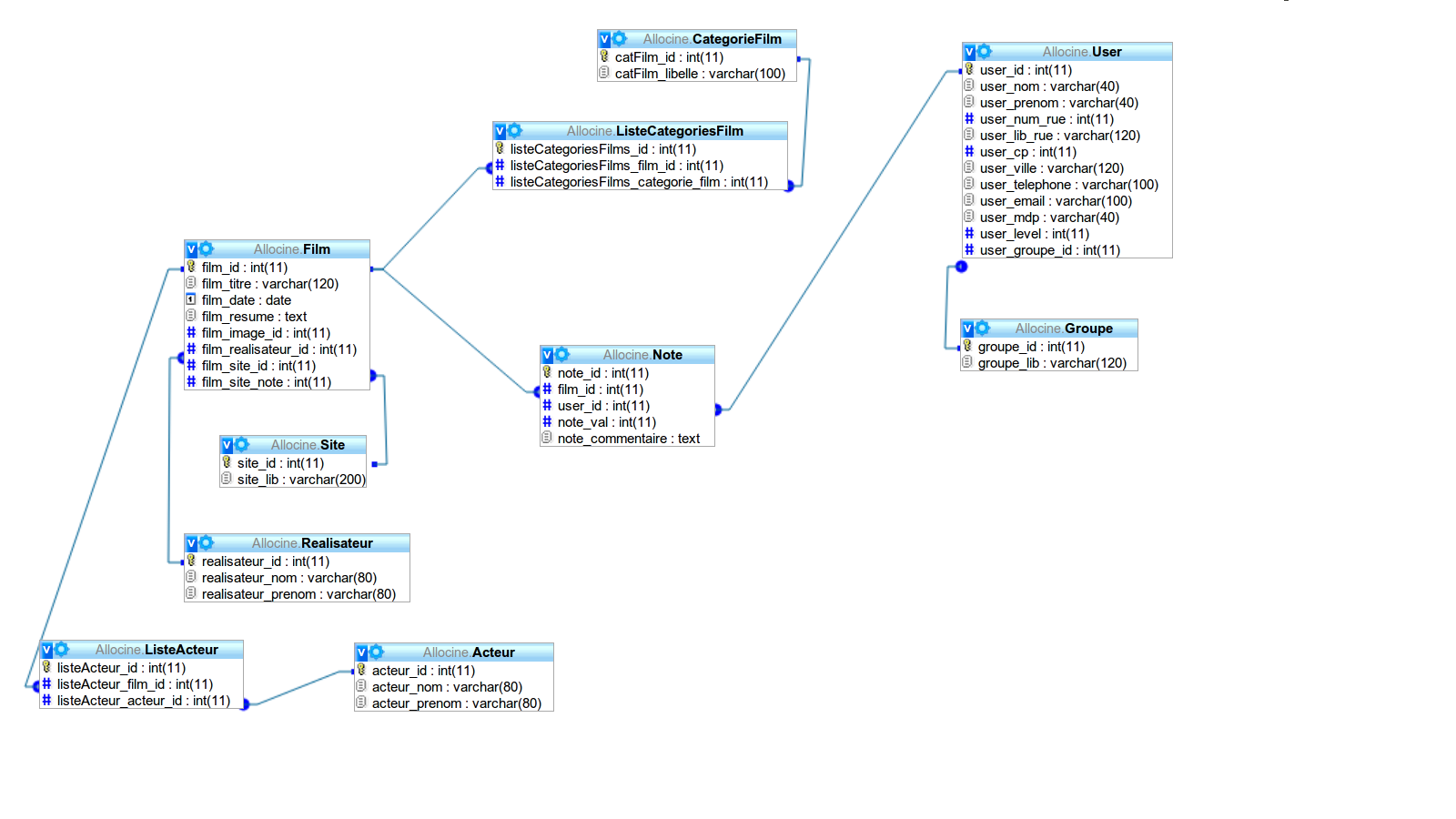
Nous avons décidé de développer la plateforme sous un modèle MVC( Modèle, Vue , Controller ). Le mvc permet d’avoir une certaine rigueur dans l’écriture du code et celui-ci permet en plus d’avoir un code structuré. Les parties qui concernent la vue sont situées dans le dossier « vue » tandis que le contrôlleur fait le lien entre les différentes vues et le modèle afin de gérer les données. Le modèle lui contient tous les objets de la plateforme.

-Utilisation d’un orm : Doctrine

Afin de faciliter l’accès aux données et de permettre une meilleure synchronisation entre la base de données et le modèle nous avons décidé d’utiliser un orm. Notre choix c’est porté sur doctrine car il est simple d’utilisation et permet de faire des requêtes complètes.

## Réalisation de la base de donnée

Dans un premier temps, il a fallu concevoir la base de données en fonction des besoins demandés.



*Schéma BDD*

## Mise en place de la partie Film

Dans un premier temps nous nous sommes concentrés sur l’accès à une liste de films et à la possibilité de voir les données du film.

* Contrainte

Il a d’abord fallu mettre en place la couche de persistence avec des pages php dao étant composées de setters et de getters. Ces méthodes permettent de récupérer facilement les données d’une table ou d’un objet du modèle. Le lien entre les tables et le modèle est gérer par l’orm, donc dans ces méthodes on fait directement appel à l’orm pour récupérer les données.

Exemple :

Si l’on veut récupérer le titre d’un film par rapport à son identifiant, on appelle la méthode getfilmTitrebyId, dans la page film\_dao.php . Cette méthode envoie une requête à l’orm afin de trouver le film qui correspond à l’identifiant. L’orm retourne ensuite un objet, ou un tableau d’objets s’il existe plusieurs films. Il suffit ensuite de récupérer les données qui nous intéressent : ici se serait donc le titre du film.

* Réalisation

Nous avons réalisé une page qui gère la recherche des films demandé par l’utilisateur, il existe 3 types de recherche :

La première est générale par rapport à l’ajout du film, la deuxième permet de rechercher par genre, la dernière est la recherche par rapport à un titre. Il est possible d’effectuer une recherche par titre à partir de n’importe quel endroit dans la plateforme.

C:\Users\etudiant\Desktop\rechercher.PNG

***Barre de recherche par titre qui se situe dans la bannière.***

L’affichage de la recherche se fait sous forme de liste on peut alors avoir déjà un aperçu des données du film, on peut voir le titre, l’année de sortie, les acteurs principaux, et l’affiche du film.



***Résultat d’une recherche.***

Une fois la recherche effectuée, il est possible de consulter la fiche d’un film en cliquant sur un bouton. On demande alors au contrôleur de nous afficher une nouvelle page fiche.php qui permet d’afficher la page du film demandé. Sur cette page toutes les informations du film sont affichées : ses informations sont récupérées via un appel à la persistance qui appel l’orm pour récupérer les données.

## Mise en place de l’inscription

* Réalisation technique

Afin de rendre l’inscription dynamique, nous avons utilisé de l’Ajax. Les informations sont envoyées à l’ORM qui crée un nouvel utilisateur dans la base de données et dans le modèle. Les différentes informations demandées sont le nom, prénom, l’adresse postale, l’adresse mail et un mot de passe. Afin de se connecter, l’utilisateur doit utiliser son adresse mail et son mot de passe.

Une fois enregistré, un utilisateur est par défaut de niveau un sur la plateforme.

## Mise en place de la gestion des utilisateurs

* Contraintes

Afin de mettre en place différents niveaux d’utilisateur, il a fallut gérer des variables de sessions permettant de connaître le niveau de chaque utilisateur. La page gestion est divisée en deux ou trois partie en fonction du niveau de l’utilisateur (la gestion de groupe, l’ajout de film et la gestion de groupe). Nous voulions que le changement de partie soit dynamique et que l’utilisateur n’est pas à recharger la page à chaque changement.

* Solutions adoptées

Afin d’accéder au différentes parties et de respecter la contrainte liée au dynamisme, nous avons opté pour la réalisation de fonctions javascript permettant d’afficher ou de cacher une partie en fonction du menu de la gestion.

Tous les utilisateurs enregistrés sur la plateforme ont accès à une gestion de leur compte et de la plateforme. La gestion diffère selon le niveau de l’utilisateur. Toutefois, tous les utilisateurs enregistrés ont le droit de noter un film une unique fois. Ils peuvent également rejoindre des groupes.

La partie groupe permet de voir les membres du groupe, on ne peut rejoindre qu’un seul groupe, par contre tous les groupes sont publique et l’utilisateur peut changer très simplement de groupe. Lorsqu’un utilisateur n’est dans aucun groupe deux choix s’offrent à lui : il a la possibilité de créer un nouveau groupe ou bien de rejoindre un groupe déjà existant. Pour que la gestion de groupes soit dynamique, nous avons également opté pour des fonctions javascript et ajax permettant de rejoindre, créer ou quitter un groupe. Une fois que l’utilisateur se situe dans un groupe, il peut avoir accès à la liste des nom et prénoms des personnes étant dans le même groupe que lui. Il peut également à tout moment quitter le groupe.



***Page de la gestion.***

## Mise en place de l’ajout de film

Il existe différentes façons d’ajouter des films sur la plateforme. On peut choisir une méthode dites manuelle qui oblige l’utilisateur à rentrer toute les données manuellement ou bien en récupérant les l’id d’un film sur le site Allociné ou IMDB.

* Contrainte

Lors de l’ajout d’un film, il existe différentes contraintes auxquels nous avons dû nous plier. En effet il faut effectuer différentes vérifications afin de savoir si le film n’est pas déjà enregistré dans la base de données ou bien si le réalisateur ou les acteurs n’existent pas déjà dans la base de données. De plus, une des contraintes majeures est de pouvoir récupérer les films sur imdb et allociné : nous avons donc dû passer par des api qui permettent de le faire. Nous avons trouvé deux api différentes: une pour Allociné et une autre pour imdb. C’est deux api retournent des données de façon différents. Une des contraintes est de pouvoir maîtriser ces api et comprendre leur fonctionnement.

* Solution adopté

D’un point de vue technique, l’ajout de film manuellement ne présente pas beaucoup de difficulté, on récupère les données soumises par l’utilisateur, après vérification, on les transmet à l’orm qui les insère dans la base de données et dans le modèle.

La partie insertion via l’identifiant d’un film d’une plateforme est plus complexe.  En effet les api retournent des données mais de types différents en fonction du site. L’api d’imdb retourne un objet tandis que l’api Allociné retourne les données au format Json. Une fois les données récupérées il faut également faire les tests sur celle-ci puis ensuite on ajoute le film et ses données.

Exemple de récupération via allociné :

1. Récupération de l’identifiant et envoi de la requête via l’api :
2. Récupération des données au format Json :



* Difficulté

La vraie difficulté était de comprendre comment manipuler les api et ensuite de sélectionner uniquement les données qui nous intéresse. Nous avons également eu des problèmes avec les ajouts de réalisateur ou d’acteurs. En effet certain films possède plusieurs réalisateurs mais la conception de notre base ne nous permettait dans avoir uniquement un seul par film. Après en avoir parlé avec notre maitre de stage nous avons décidé de sélectionner uniquement un seul réalisateur. Nous avons également rencontré des problèmes pour l’ajout des noms des acteurs n’ayant prévue qu’un acteur possède uniquement un prénom ou nom. Mais certains acteurs peuvent posséder un nom composé tel que Tommy Lee Jones : nous avons donc tronqué son nom.

## Mise en place de la liste de favoris

Afin de rendre la plateforme plus communautaire, nous avons mis en place une liste de films favoris pour chaque utilisateur enregistré.

* Contrainte

Il a fallut modifier la base de données pour que chaque utilisateur puisse posséder une liste de films favoris. De plus, afin de mettre en place un aspect communautaire, il a fallut mettre en place un système pour que les autres utilisateurs aient également accès à la liste de chacun. L’utilisateur devait également pouvoir gérer sa liste de favoris de manière simple.

* Solution adoptée

Lorsqu’un utilisateur est connecté et se situe sur une page d’un film, il a alors la possibilité d’ajouter ce film à sa liste de films favoris en cliquant sur un bouton « plus » situé en haut a droite. Dans le cas où le film figure déjà dans la liste des films favoris de l’utilisateur nous avons remplacé le bouton « plus » par un bouton « moins » signifiant le retrait de ce film de la liste de favoris. Pour ce faire, il a fallu faire des tests à chaque fois qu’un utilisateur charge une fiche d’informations d’un film, on regarde alors en fonction de l’id du film et l’id de l’utilisateur si le film est parmi les favoris ou non.

Afin de mettre en place la gestion des favoris et un accès simple, nous avons ajouté un lien dans l’entete de la plateforme afin que l’utilisateur puisse rejoindre sa page des films favoris à tout moment. Dans la page de gestion des favoris, l’utilisateur peut voir l’affiche du film , le titre de celui-ci et il à également la possibilité de supprimer le film de la liste de ses favoris. L’orsque l’utilisateur clique sur le titre, il accède directement à la fiche d’informations du film

Pour voir la liste des films favoris des autres membres de la communauté, nous avons mis en place dans la gestion des groupe, la liste des membres de notre groupe avec leur nom, leur prénom et un lien menant directement à la liste des favoris si celle-ci existe.

.

***Gestion des favoris.***

## Mise en place de la notation d’un film

Pour répondre au besoin de notation du film, nous avons mis en place dans l’affichage du film un système de notation par étoile. Le système de notation est compris entre zéro et cinq, zéro étant la note la plus faible et cinq la note la plus forte.

* Contrainte

Une des contraintes pour la notation d’un film est la possibilité de noter une seule fois un film. En effet, seuls les utilisateurs enregistrés doivent pouvoir noter un film. Une autre contrainte est que nous souhaitions avoir une notation dynamique. Nous avons dû effectuer des recherches afin de trouver une notation par étoile : une fois que l’utilisateur clique sur une étoile, il doit pouvoir voir la note qu’il à attribué et ne doit pas pouvoir revoter.

* Solution adoptée

La solution adoptée pour le problème d’unique notation est d’effectuer une vérification. On exécute une requête avec l’id du film et celui de l’utilisateur afin de savoir si celui-ci a déjà noté le film. Dans le cas où l’utilisateur a déjà noté, le film on affiche la note qu’il a attribuée. Dans le cas où l’utilisateur n’a pas noté le film, on affiche les étoiles de notations. Afin de palier au problème d’effet dynamique, nous avons mis en place un script javascript et ajax. Le script javascript donne l’effet dynamique des étoiles lorsque l’utilisateur clique sur une étoile : ce qui correspond à la notation qu’il souhaite donner au film. Une requête ajax retourne la note donnée par l’utilisateur et l’affiche ensuite dans la page.

C:\Users\etudiant\Desktop\note etoile.PNG

*Système de notation dynamique*

C:\Users\etudiant\Desktop\votre note.PNG

*Note attribuée par l'utilisateur*

## Mise en place des statistiques

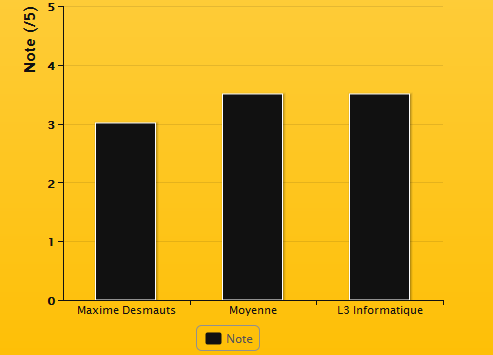
Le système de statistique permet de voir la notation d’un film.

* Contrainte

Les contraintes de la mise en place des statistiques étaient d’un point de vue graphique. En effet il a d’abord fallut trouver une bibliothèque externe permettant de générer des statistiques. Dans un deuxième temps, il a fallu maîtriser cette bibliothèque et convertir les données de la base de données pour qu’elles soient acceptées par la bibliothèque. Il fallait également gérer des statistiques par groupe.

* Solution

Les statistiques sont accessible à partir de la liste de recherche des films, un lien situé en dessous de la note du film permet d’accéder à la page des statistiques .Nous avons choisis d’utiliser Highchart : un frameworks javascript pour générer des diagrammes. Afin de créer les diagrammes, Highchart récupère les données sous le format Json et génère les diagrammes correspondant. Un utilisateur peut avoir accès au maximum à 3 diagrammes, par défaut seul deux diagrammes sont affichés. Les diagrammes correspondent à la notation générale du film et à la note de l’utilisateur. Enfin, si l’utilisateur est dans un groupe, on rajoute la note moyenne des utilisateurs qui appartiennent au groupe : pour ce dernier, nous avons dû récupérer tous les membres d’un groupe qui ont noté ce film et ensuite en faire la moyenne.



***Diagramme de notation pour un film.***

## CSS et design du site

* Contrainte

La plateforme à été conçut avec un design bien particulier. Afin de mettre en place le design nous avons utilisée du CSS 3. Conscient que tous les navigateurs ne le prennent pas encore en compte nous avons donc dût écrire un style qui devait faire le compromis entre tous les navigateurs.

* Réalisation

Nous avons donc réalisé un design simple et clair. Nous avons également mit en place dans la partie gestion, la possibilité de changer le design du site. Nous avons mit un choix de 5 couleurs à l’utilisateur. En cliquant sur une des couleurs, celle-ci deviendra la couleur dominante de la plateforme.

D’un point de vue technique, il existe cinq feuilles de styles qui possèdent toutes les cinq des couleurs différentes, en fonction du style que choisit l’utilisateur la feuille de style qui correspond est sélectionnée.

## Gestion des tables

* Difficulté

Un des besoins était de pouvoir modifier les tables de la base de données à partir de la plateforme. La difficulté principale était de faire en sorte que dans notre code source lors d’une modification du nom d’une table par exemple, les requêtes soient modifiées également. Cela nous a vite parut très compliqué à réaliser en si peu de temps. Nous avons donc dû abandonner cette idée.

# 4-Tests effectués

Afin de régler les derniers problèmes, nous avons dû effectuer des séries de test. Nous avons entre-autre rencontré des problèmes lors de l’ajout de film et avec imdb ou allociné. En effet, certains acteurs ou films étaient identifiés comme nul dans la base de données. Cela engendrait des erreurs lors de la consultation de la fiche détaillée du film car nous ne prenions pas en compte le fait que certains attributs (réalisateur, acteurs, ..) étaient inexistants.

Les deux grands rails de test s’effectuaient sur :

1 – Partie utilisateur

Dans cette partie, les tests se visualisaient surtout au niveau de la base de données. Exemple : lors de l’inscription, on regarde si l’utilisateur a bien été ajouté. Ensuite, il y’a la partie modification des droits d’utilisateurs via le super utilisateur. En sachant que l’on recharge la page, on peut voir directement la modification effectuée. Enfin, la suppression d’un utilisateur se fait via le super administrateur donc idem que la modification du niveau, on peut voir la réussite de la suppression au chargement de la page.

2 – Partie film

Cette partie est un peu plus complexe que la partie utilisateur car pour l’ajout des films via les sites imdb et allociné, nous passions par des api téléchargées sur le web (voir les adresses des sites web dans la partie bibliographie). Pour l’ajout des films, nous faisions des « try – catch » (gestionnaire d’exceptions) ce qui nous permettait de détecter les éventuelles erreurs. Si l’ajout était un succès, nous pouvions le voir dans la liste des films.

Pour conclure sur cette rubrique de tests, nous avons vérifié la plupart du temps via la base de données ou alors en effectuant des tests d’affichage pour vérifier les valeurs de variables.

# 5-Evolutions possible de la plateforme

La plateforme, comme cela l’était demandé, peut être évoluée ou améliorée. Nous aurions aimé mettre en place un système de commentaires pour les films, mais par manque de temps, nous n’avons pas pu le réaliser. Evidemment, il existe une multitude de choses qui peut être amélioré ou évolué tel que la gestion des statistiques, etc…

Il serait également possible de mettre en place la partie de modification des tables de la base de données. Par un manque de temps, nous n’avons pas pu mettre en place cette partie. Le gros problème étant que cette modification influait sur notre code : c’est-à-dire que si l’on supprime un champ d’une table, alors on doit modifier cela sur nos pages php ainsi que le modèle et la dao (qui effectue les requêtes sql). C’est bien évidemment par manque de temps donc que cet ajout n’était pas réalisable.

Dans un dernier temps, on pourrait diversifier le site en rajoutant une partie livre par exemple, ou alors des jeux vidéo : ce qui, en sommes, n’est pas très compliquer car ce serait le même modèle que pour les films : des tables similaires, une interface identique… Il faudrait juste trouver des sites d’import ainsi qu’un api correspondant.

# 6-Conclusion

Ce stage nous a permis d’améliorer nos compétences en développement web, mais également de nous rendre compte de la difficulté de certain projet. Il nous a fallu une certaine rigueur dans le travail, la contrainte d’avoir un site évolutif et facile d’utilisation nous imposant cette rigueur.

Cette expérience a été pour nous une bonne façon de pouvoir réutiliser la théorie vue lors de nos enseignements dans un contexte concret au sein d’une vraie application. Evidemment, nous avons rencontré beaucoup de problèmes lors de notre réalisation.

La partie d’ajout de films via des api nous a apporté d’autres connaissances qui pourraient nous être utile dans le futur.

# 7- Bibliographie

APIs :

* IMDB

*+* [*http://www.imdbapi.com/*](http://www.imdbapi.com/)

*+* [*http://api.themoviedb.org/2.1/*](http://api.themoviedb.org/2.1/)

*+* [*http://projects.izzysoft.de/trac/imdbphp*](http://projects.izzysoft.de/trac/imdbphp)

* Allociné

*+* [*http://krittik.blogy.fr/recuperation-de-donnees-c83293*](http://krittik.blogy.fr/recuperation-de-donnees-c83293)

*+* [*https://github.com/etienne-gauvin/api-allocine-helper*](https://github.com/etienne-gauvin/api-allocine-helper)

Frameworks :

* Dojo Toolkit

*+* [*http://dojotoolkit.org/*](http://dojotoolkit.org/)

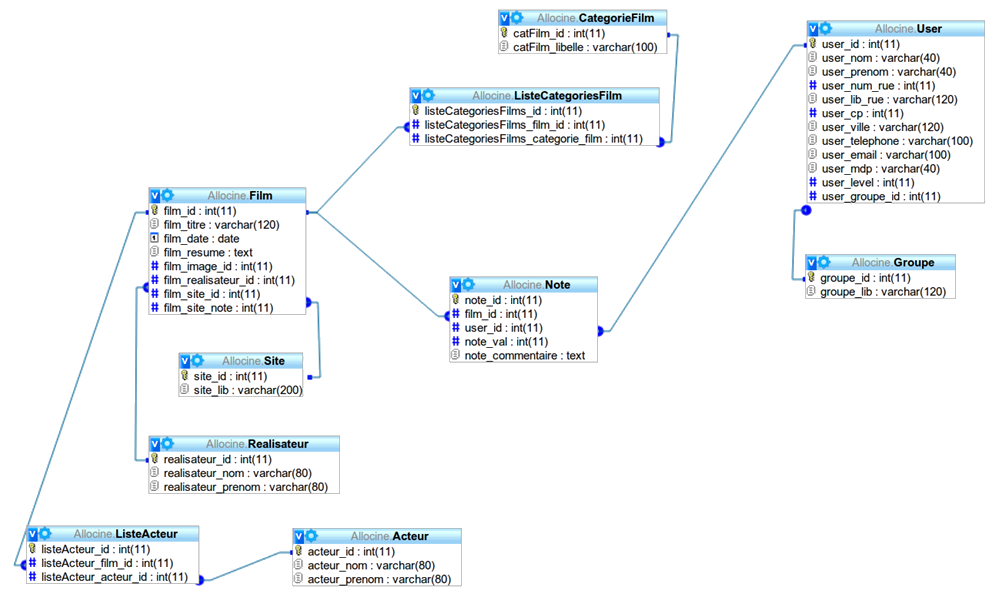
* HighCharts

*+* [*http://www.highcharts.com/*](http://www.highcharts.com/)

Tutoriels Css :

*+* [*http://www.css3create.com/*](http://www.css3create.com/)

# 8- Annexes



Explication de la base de données :

* Un **film** peut posséder plusieurs catégories (**listeCatégoriesFilm**) ainsi que plusieurs acteurs (**listeActeur**), peut être issue d’un **site** et possède un **réalisateur**
* Un **user** peut appartenir à un **groupe** et peut noter (**note**) un **film**

Dictionnaire des données :

## ACTEUR

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Colonne** | **Type** | **Null** | **Défaut** | **Commentaires** |
| acteur\_id | int(11) | Non |  |  |
| acteur\_nom | varchar(80) | Oui | *NULL* |  |
| acteur\_prenom | varchar(80) | Oui | *NULL* |  |

## CATEGORIEFILM

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Colonne** | **Type** | **Null** | **Défaut** | **Commentaires** |
| catFilm\_id | int(11) | Non |  |  |
| catFilm\_libelle | varchar(100) | Oui | *NULL* |  |

## FILM

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Colonne** | **Type** | **Null** | **Défaut** | **Commentaires** |
| film\_id | int(11) | Non |  |  |
| film\_titre | varchar(120) | Oui | *NULL* |  |
| film\_date | date | Oui | *NULL* |  |
| film\_resume | text | Non |  |  |
| film\_image\_id | varchar(40) | Oui | *NULL* |  |
| film\_realisateur\_id | int(11) | Oui | *NULL* |  |
| film\_site\_id | int(11) | Oui | *NULL* |  |
| film\_site\_note | int(11) | Oui | *NULL* |  |

## FILM\_FAVORIS

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Colonne** | **Type** | **Null** | **Défaut** | **Commentaires** |
| film\_favoris\_id | int(11) | Non |  |  |
| film\_id | int(11) | Non |  |  |
| user\_id | int(11) | Non |  |  |

## GROUPE

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Colonne** | **Type** | **Null** | **Défaut** | **Commentaires** |
| groupe\_id | int(11) | Non |  |  |
| groupe\_lib | varchar(120) | Oui | *NULL* |  |

## LISTEACTEUR

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Colonne** | **Type** | **Null** | **Défaut** | **Commentaires** |
| listeActeur\_id | int(11) | Non |  |  |
| listeActeur\_film\_id | int(11) | Oui | *NULL* |  |
| listeActeur\_acteur\_id | int(11) | Oui | *NULL* |  |

## LISTECATEGORIESFILM

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Colonne** | **Type** | **Null** | **Défaut** | **Commentaires** |
| listeCategoriesFilms\_id | int(11) | Non |  |  |
| listeCategoriesFilms\_film\_id | int(11) | Non |  |  |
| listeCategoriesFilms\_categorie\_film | int(11) | Non |  |  |

## LISTERECOMPENSES

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Colonne** | **Type** | **Null** | **Défaut** | **Commentaires** |
| listeRecompense\_id | int(11) | Non |  |  |
| listeRecompense\_film\_id | int(11) | Oui | *NULL* |  |
| listeRecompense\_recompense\_id | int(11) | Oui | *NULL* |  |

## NOTE

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Colonne** | **Type** | **Null** | **Défaut** | **Commentaires** |
| note\_id | int(11) | Non |  |  |
| film\_id | int(11) | Non |  |  |
| user\_id | int(11) | Oui | *NULL* |  |
| note\_val | int(11) | Non |  |  |
| note\_commentaire | text | Oui | *NULL* |  |

## REALISATEUR

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Colonne** | **Type** | **Null** | **Défaut** | **Commentaires** |
| realisateur\_id | int(11) | Non |  |  |
| realisateur\_nom | varchar(80) | Oui | *NULL* |  |
| realisateur\_prenom | varchar(80) | Oui | *NULL* |  |

## RECOMPENSE

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Colonne** | **Type** | **Null** | **Défaut** | **Commentaires** |
| recompense\_id | int(11) | Non |  |  |
| recompense\_lib | varchar(120) | Oui | *NULL* |  |

## SITE

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Colonne** | **Type** | **Null** | **Défaut** | **Commentaires** |
| site\_id | int(11) | Non |  |  |
| site\_lib | varchar(200) | Oui | *NULL* |  |

## USER

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Colonne** | **Type** | **Null** | **Défaut** | **Commentaires** |
| user\_id | int(11) | Non |  |  |
| user\_nom | varchar(40) | Non |  |  |
| user\_prenom | varchar(40) | Non |  |  |
| user\_num\_rue | int(11) | Oui | *NULL* |  |
| user\_lib\_rue | varchar(120) | Oui | *NULL* |  |
| user\_cp | int(11) | Oui | *NULL* |  |
| user\_ville | varchar(120) | Oui | *NULL* |  |
| user\_telephone | varchar(100) | Oui | *NULL* |  |
| user\_email | varchar(100) | Non |  |  |
| user\_mdp | varchar(40) | Non |  |  |
| user\_level | int(11) | Non |  |  |
| user\_groupe\_id | int(11) | Oui | *NULL* |  |