Raphaël Lopes

CFPT-INFORMATIQUE

Avril 2019

Documentation TECHNIQUE

TALES OF THE TAVERN

TRAVAIL PRATIQUE INDIVIDUEL 2019

Experts : M. Pascal COURT, M. Borys FOLOMIETOW

Maitre d’apprentissages : Mme Katia MOTA STROPPOLO

Table des matières

[1 Table des versions 3](#_Toc6239060)

[2 Introduction 4](#_Toc6239061)

[2.1 Généralités 4](#_Toc6239062)

[3 Cahier des charges 4](#_Toc6239063)

[3.1 Objectif 4](#_Toc6239064)

[3.2 Fonctionnalités 4](#_Toc6239065)

[4 Planification 5](#_Toc6239066)

[4.1 Planification prévue 5](#_Toc6239067)

[4.2 Planification effective 5](#_Toc6239068)

[5 Méthodologie 5](#_Toc6239069)

[5.1 Planification en 6 étapes 5](#_Toc6239070)

[5.2 Développement agile / BACKLOG 6](#_Toc6239071)

[6 Analyse organique 7](#_Toc6239072)

[6.1 Base de données 7](#_Toc6239073)

[6.1.1 Informations 7](#_Toc6239074)

[6.2 MCD 8](#_Toc6239075)

[6.3 MLD 8](#_Toc6239076)

[6.4 Dictionnaire de données 9](#_Toc6239077)

[6.5 Description 11](#_Toc6239078)

[6.6 Architecture 11](#_Toc6239079)

[Tales of the tavern/index.php 12](#_Toc6239080)

[Tales of the tavern/Vue 12](#_Toc6239081)

[Tales of the tavern/Controleur 13](#_Toc6239082)

[Tales of the tavern/Model 14](#_Toc6239083)

[Tales of the tavern/bootstrap-4.3.1-dist 14](#_Toc6239084)

[Tales of the tavern/Img 14](#_Toc6239085)

[6.7 Analyse d’une fonctionnalité principale 15](#_Toc6239086)

[7 Test et plan de test 15](#_Toc6239087)

[8 Conclusion 15](#_Toc6239088)

[8.1 Objectifs 15](#_Toc6239089)

[8.2 Difficultés rencontrées 15](#_Toc6239090)

[8.3 Apprentissages 15](#_Toc6239091)

[8.4 Remerciements 15](#_Toc6239092)

[9 Bibliographie 16](#_Toc6239093)

[10 Annexes 16](#_Toc6239094)

# Table des versions

|  |  |  |
| --- | --- | --- |
| Date | Modification | Version |
| 02.04.19 | Création du document | 0.1 |
| 02.04.19 | Page de garde, Introduction, Planification | 0.2 |
| 03.04.19 | Base de données | 0.3 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

# Introduction

## Généralités

Dans le cadre du travail pratique individuel, j’ai dû réaliser un projet dans le domaine de l'informatique, qui consiste d’une application web.

La fonction principale de l’application est que des utilisateurs (connecté à un compte) puissent mettre en ligne des histoires avec une image et que par la suite d’autres utilisateurs peuvent les regarder et le noter sur quatre critères. Enfin, il y a une dernière note qui est la moyenne des 4

1. Styles.
2. Histoire
3. Orthographe
4. Originalité

# Cahier des charges

## Objectif

Développer un site web permettant de s’y connecter avec un compte, de pouvoir créer des histoires et de noter des histoires que les autres utilisateurs ont créées.

## Fonctionnalités

Les fonctionnalités principales sont :

* Créer un compte / se connecter
* Créer une histoire / afficher l’histoire
* Pouvoir noter une histoire
* Rechercher une histoire
* Pouvoir voir les histoires les plus récentes (en date) et voir les mieux noter

Une histoire contient

* Un titre
* Un auteur
* Une date de création
* Une histoire
* Une note en dans 4 diffèrent critère :
  + Style
  + Histoire
  + Orthographe
  + Originalité
* Une note générale qui est la moyenne des 4.
* La catégorie de l’histoire
* Une image (s’il n’y a pas d’image alors on affiche celle de la catégorie)

L’application prévoit 3 types d’utilisateurs

* L’anonyme : c’est un utilisateur qui n’est pas connecté. Il peut seulement lire des histoires et les noter.



* Le membre : c’est un utilisateur qui s’est connecté au préalable. Il a les mêmes droits que « l’anonyme », mais peut en plus, afficher et gérer depuis la page d’accueil une liste d’histoire favorite et, il a une page personnelle permettant de gérer ses informations
  + Nom d’utilisateur (lire, modifier).



* + Email
  + Mot de passe
  + Note générale
  + Toutes les histoires créées par l’utilisateur
* L’administrateur : en plus des droits ci-dessus l’administrateur doit gérer la base de données avec l’interface d’administration de la base de données et de maintenir le code source de l’application.

# Planification

## Planification prévue

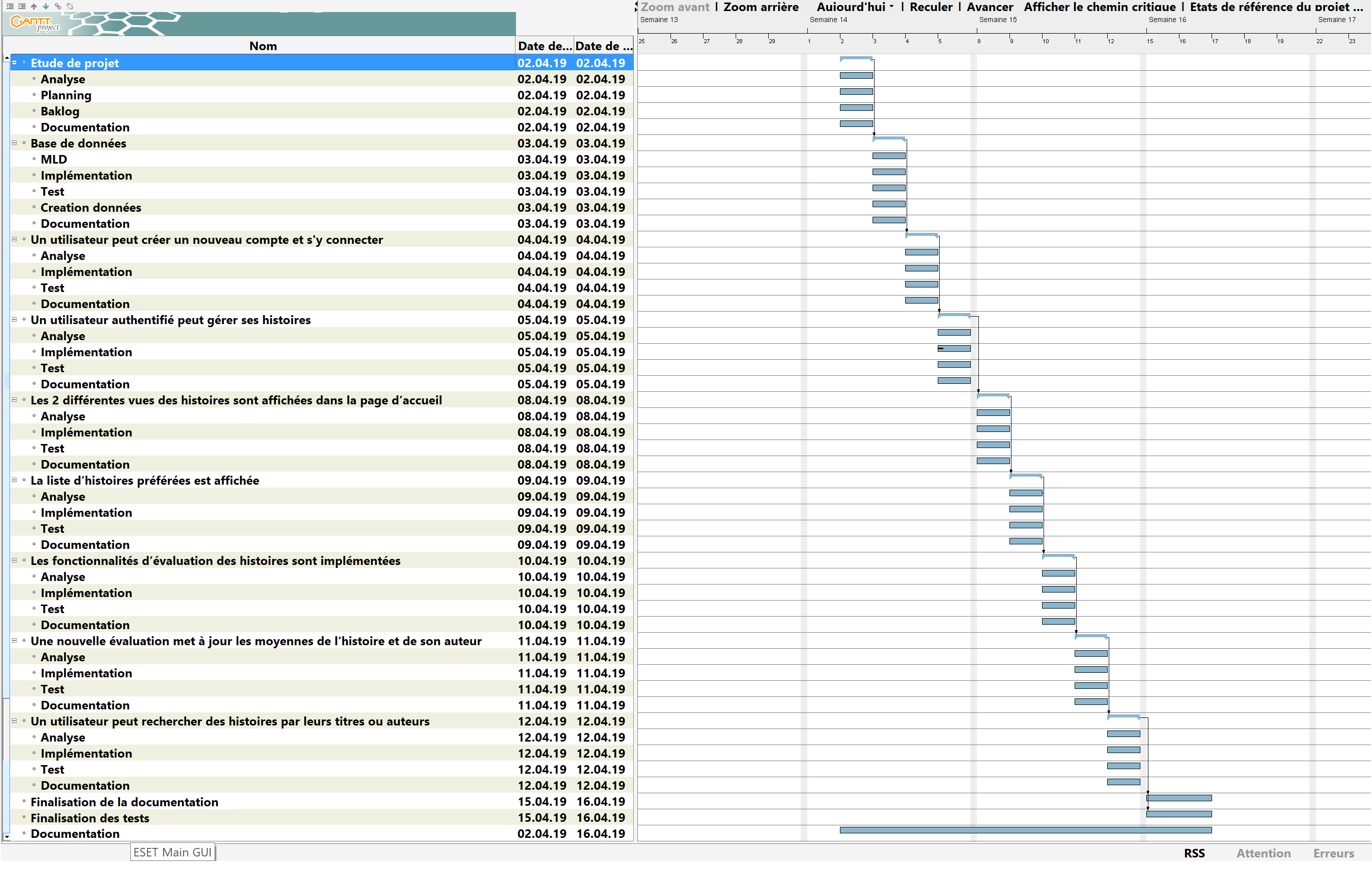


Figure 1 : Planning prévue

## Planification effective

On peut voir qu’entre le planning réelle et effectif…

# Méthodologie

## Planification en 6 étapes

La méthodologie en 6 étapes aide à exécuter, organiser et faire avancer le travail

Elle se divise comme son nom l’indique en 6 étapes :

* Informer : consiste à comprendre et enregistrer la tâche pour se faire une idée et bien cerner le sujet et avoir le maximum d’information
* Planifier : cette étape consiste à répartir les tâches, choisir les tâches et estimer le temps nécessaire.
* Décider : lorsque l’information et de la planification sont faites, il faut prendre une décision.
* Réaliser : cette étape est la partie principale d’une tâche, mais il faut néanmoins respecter le plan que nous avons créé à la planification et ne pas le modifier (si possible)
* Contrôler : chaque travail fait doit être contrôlé, il faut donc le relire, comparer avec le cahier des charges, mesurer et estimer.
* Évaluer : enfin, il faut évaluer tout ce que ne nous avons fait pour savoir si le travail effectuer est bon ou non. Savoir ou avons-nous perdu du temps, fait des erreurs, si tout c’est bien passé, etc.

La méthodologie a été appliquée dans la planification dans le GANTT

J’ai utilisé cette méthodologie dans la panification dans le GANTT pour chaque tâche.

## Développement agile / BACKLOG

Pour chaque critère j’ai décider de la prendre et la transformer en user stories. Ensuite chaque jours je fais une sorte de sprint avec l’user stories du jours et je l’implémente. J’ai utilisé cette méthodologie tout le long du développement.

Tableau 1 : User stories

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ID | En tant que … | Je souhaite… | Pour | Resultat obtenue | Statut |
| A14 | Anonyme | créer un nouveau compte et pourvoir m'y connecter | Avoir une page "création du compte" fonctionnel et une page de "login" qui permettent de se connecter |  | À faire |
| A15 | Admin/Membre | avoir accès à ma page de gestion d'histoire | Avoir une page "mon compte" qui affiche les informations du compte ( du nom d'utilisateur aux histoires qu'il a créé). |  | À faire |
| A16 | Anonyme/Admin/Membre | Avoir 2 tris sur la page accueille 1-moyenne de notes (tri croissant) 2-Les histoires plus récentes (tri décroissant) | Pouvoir afficher les histoires sur la page d'accueil avec 2 tris différents. 1-moyenne de notes (tri croissant) 2-Les histoires plus récentes (tri décroissant) |  | À faire |
| A17 | Admin/Membre | avoir mes histoires que j'ai marquées en favoris sur la page accueille et pouvoir les supprimer | afficher sur la page principale des utilisateurs identifier leur histoire favorite et pouvoir les supprimer |  | À faire |
| A18 | Anonyme/Admin/Membre | pouvoir évaluer une histoire selon le style, l'histoire, orthographe et l'originalité. | Permettre à n'importe quel utilisateur de pouvoir noter une histoire |  | À faire |
| A19 | Anonyme/Admin/Membre | que lorsque j'évalue une histoire sa note et la note de l'auteur change | Que les moyenne des histoire / auteur change dans le temps |  | À faire |
| A20 | Anonyme/Admin/Membre | pouvoir chercher une histoire par son titre ou par son auteur | Permettre aux utilisateurs de chercher une histoire avec le titre |  | À faire |

# Analyse organique

## Base de données

### Informations

PhpMyAdmin : version 17.0

PHP : version 5.6.30

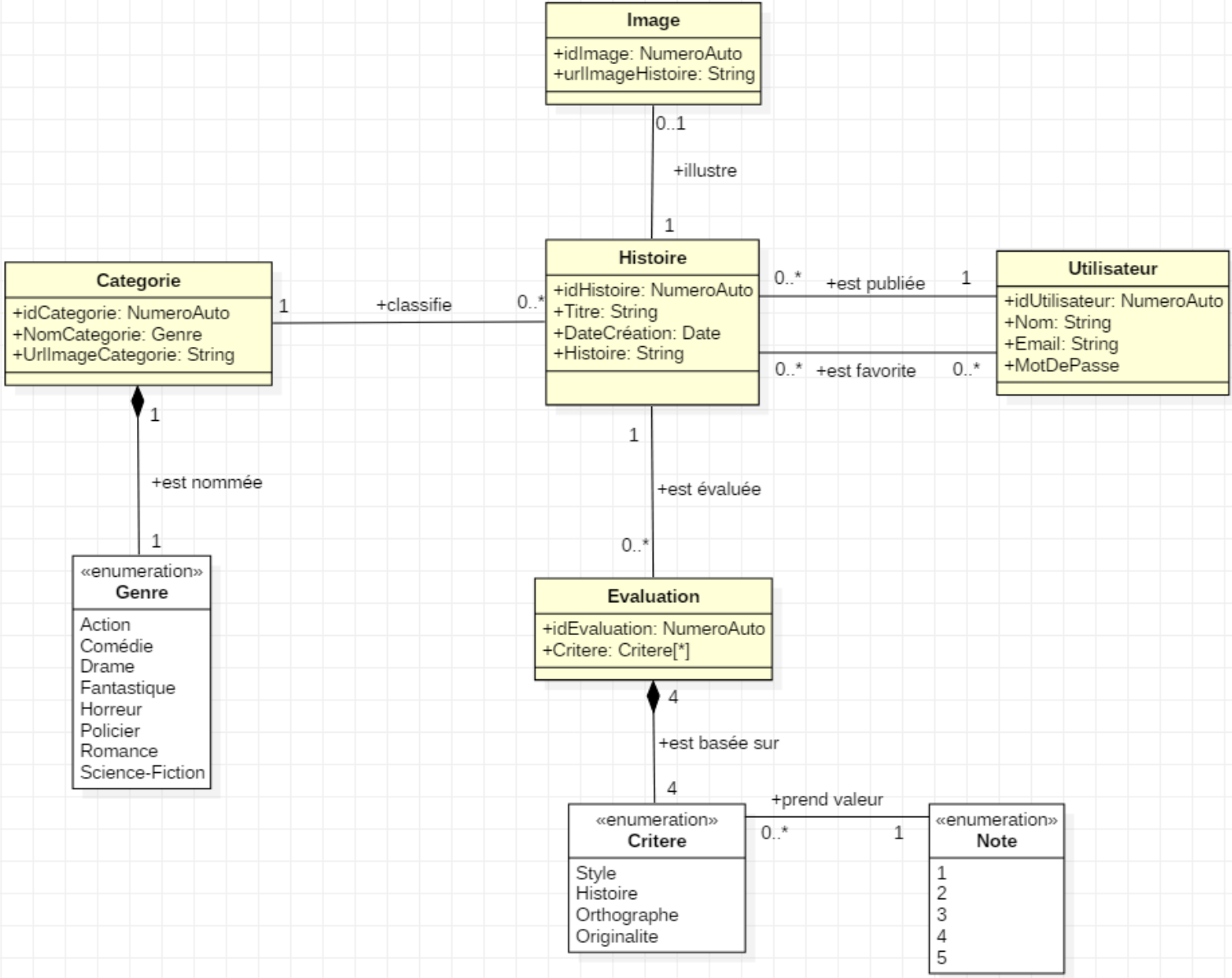
Apache : version 2.4.25

MySQL : version 5.7.17

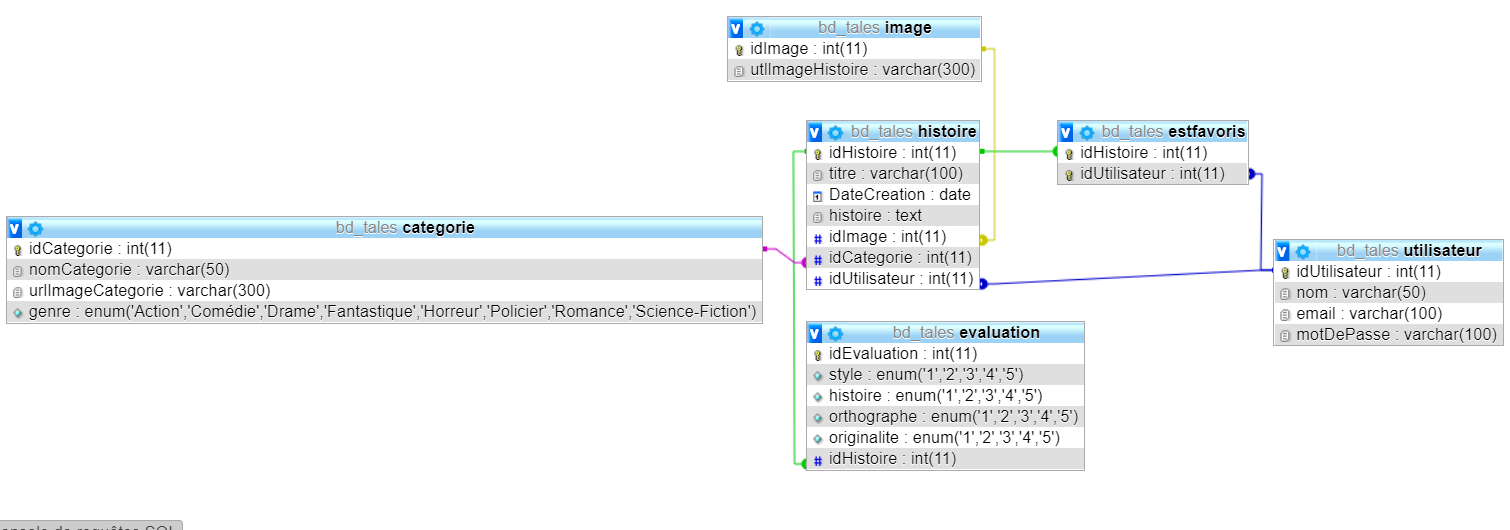
Moteur de base de données : InnoDB

Interclassement : utf8mb4\_general\_ci

## MCD



## MLD



## Dictionnaire de données

**catégorie**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Colonne | Type | Null | Valeur par défaut | Est relié à | Commentaires | MIME |
| idCategorie (Primaire) | int(11) | Non |  |  |  |  |
| nomCategorie | varchar(50) | Non |  |  |  |  |
| urlImageCategorie | varchar(100) | Non |  |  |  |  |
| genre | enum('Action', 'Comédie', 'Drame', 'Fantastique', 'Horreur', 'Policier', 'Romance', 'Science-Fiction') | Non |  |  |  |  |

Index

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Nom de l'index | Type | Unique | Compressé | Colonne | Cardinalité | Interclassement | Null | Commentaire |
| PRIMARY | BTREE | Oui | Non | idCategorie | 3 | A | Non |  |

**estfavoris**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Colonne | Type | Null | Valeur par défaut | Est relié à | Commentaires | MIME |
| idHistoire (Primaire) | int(11) | Non |  | histoire -> idHistoire |  |  |
| idUtilisateur (Primaire) | int(11) | Non |  | utilisateur -> idUtilisateur |  |  |

Index

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Nom de l'index | Type | Unique | Compressé | Colonne | Cardinalité | Interclassement | Null | Commentaire |
| PRIMARY | BTREE | Oui | Non | idHistoire | 3 | A | Non |  |
| idUtilisateur | 4 | A | Non |
| idHistoire | BTREE | Non | Non | idHistoire | 3 | A | Non |  |
| idUtilisateur | BTREE | Non | Non | idUtilisateur | 2 | A | Non |  |

**evaluation**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Colonne | Type | Null | Valeur par défaut | Est relié à | Commentaires | MIME |
| idEvaluation (Primaire) | int(11) | Non |  |  |  |  |
| style | enum('1', '2', '3', '4', '5') | Non |  |  |  |  |
| histoire | enum('1', '2', '3', '4', '5') | Non |  |  |  |  |
| orthographe | enum('1', '2', '3', '4', '5') | Non |  |  |  |  |
| originalite | enum('1', '2', '3', '4', '5') | Non |  |  |  |  |
| idHistoire | int(11) | Non |  | histoire -> idHistoire |  |  |

Index

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Nom de l'index | Type | Unique | Compressé | Colonne | Cardinalité | Interclassement | Null | Commentaire |
| PRIMARY | BTREE | Oui | Non | idEvaluation | 2 | A | Non |  |
| idHistoire | BTREE | Non | Non | idHistoire | 2 | A | Non |  |

**histoire**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Colonne | Type | Null | Valeur par défaut | Est relié à | Commentaires | MIME |
| idHistoire (Primaire) | int(11) | Non |  |  |  |  |
| titre | varchar(100) | Non |  |  |  |  |
| dateCreation | date | Non |  |  |  |  |
| histoire | text | Non |  |  |  |  |
| idImage | int(11) | Oui | *NULL* | image -> idImage |  |  |
| idCategorie | int(11) | Non |  | categorie -> idCategorie |  |  |
| idUtilisateur | int(11) | Non |  | utilisateur -> idUtilisateur |  |  |

Index

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Nom de l'index | Type | Unique | Compressé | Colonne | Cardinalité | Interclassement | Null | Commentaire |
| PRIMARY | BTREE | Oui | Non | idHistoire | 2 | A | Non |  |
| idImage | BTREE | Non | Non | idImage | 2 | A | Oui |  |
| idCategorie | BTREE | Non | Non | idCategorie | 2 | A | Non |  |
| idUtilisateur | BTREE | Non | Non | idUtilisateur | 2 | A | Non |  |

**image**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Colonne | Type | Null | Valeur par défaut | Est relié à | Commentaires | MIME |
| idImage (Primaire) | int(11) | Non |  |  |  |  |
| urlImageHistoire | varchar(100) | Non |  |  |  |  |

Index

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Nom de l'index | Type | Unique | Compressé | Colonne | Cardinalité | Interclassement | Null | Commentaire |
| PRIMARY | BTREE | Oui | Non | idImage | 4 | A | Non |  |

**utilisateur**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Colonne | Type | Null | Valeur par défaut | Est relié à | Commentaires | MIME |
| idUtilisateur (Primaire) | int(11) | Non |  |  |  |  |
| Nom | varchar(50) | Non |  |  |  |  |
| Email | varchar(100) | Non |  |  |  |  |
| motDePasse | varchar(100) | Non |  |  |  |  |

Index

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Nom de l'index | Type | Unique | Compressé | Colonne | Cardinalité | Interclassement | Null | Commentaire |
| PRIMARY | BTREE | Oui | Non | idUtilisateur | 3 | A | Non |  |
| email | BTREE | Oui | Non | email | 3 | A | Non |  |

[Ouvrir une nouvelle fenêtre phpMyAdmin](http://127.0.0.1/eds-modules/phpmyadmin470x181106132508/db_datadict.php?db=bd_tales&table=utilisateur&server=1&target=)

## Description

## Architecture

Le site web se divise en trois parties la vue, le contrôleur et le modèle (découpage MVC) lorsque l’utilisateur fait une action sur la vue, elle appelle le contrôleur qui va traiter l’information, si le contrôleur a besoin des informations stocker dans la base alors, le contrôleur fait appelle au modèle qui va lui retourner les informations de la base qui transforme ces informations en information utilisable pour la vue.

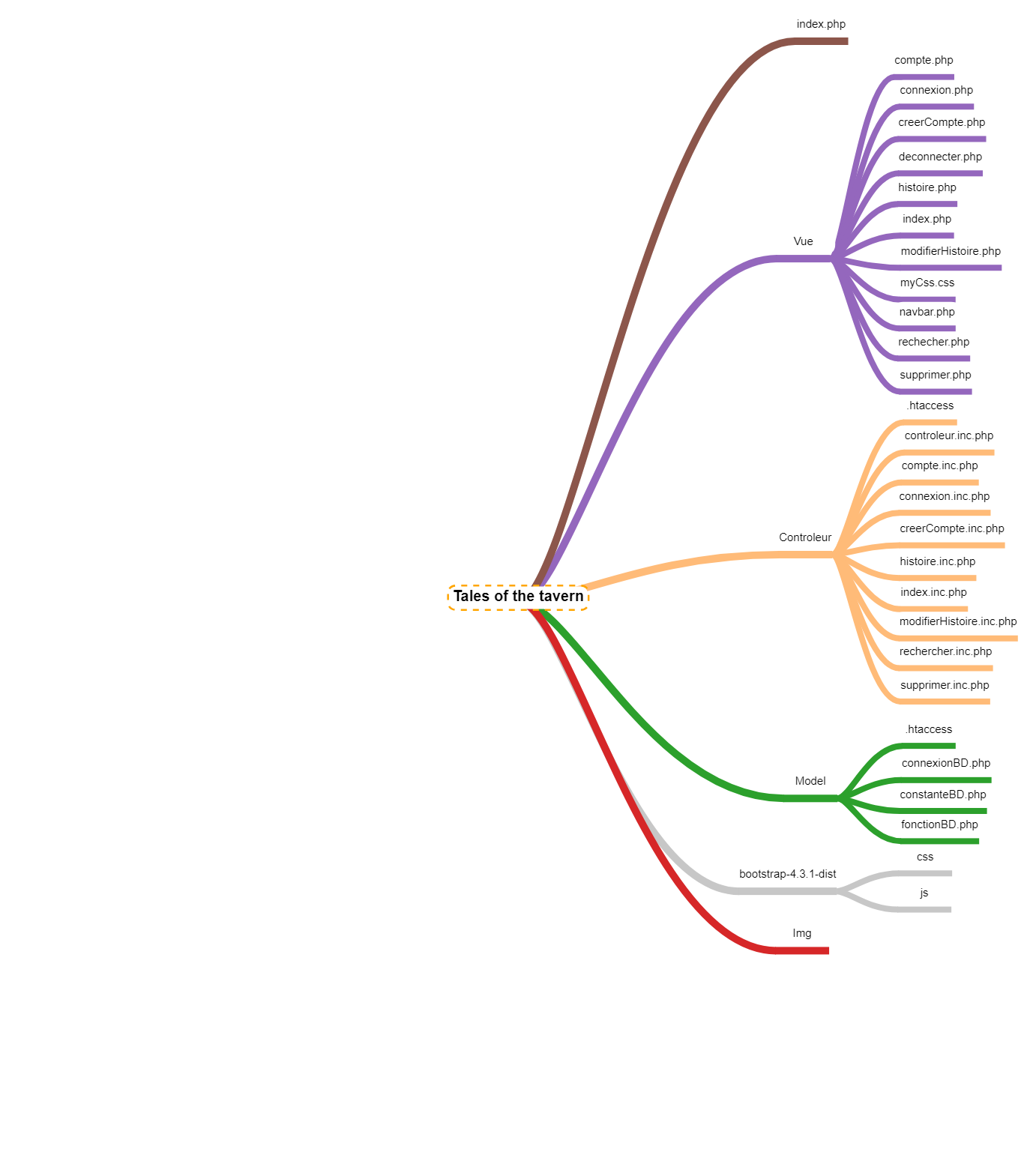


Figure 2 : Architecture du site web

### Tales of the tavern/index.php

Cette page redirige l’utilisateur vers Vue/index.php

### Tales of the tavern/Vue

Ce dossier ne fait qu’afficher les informations et en cas d’action il envoie les données à traiter au diffèrent contrôleur.

* compte.php : Cette page permet d’afficher les champs pour la modification des informations de l’utilisateur de plus il affiche les histoires créer par l’utilisateur avec 2 boutons : un pour supprimer l’autre modifier. Elle redirige tous les utilisateurs non connecter vers la page login.php.
* connexion.php : cette page permet d’afficher les champs à l’utilisateur pour se connecter. Elle redirige l’utilisateur vers la page index.php s’il est déjà connecté.
* creerCompte.php : Comme pour la page connexion.php si l’utilisateur est déjà connecté alors la page le redirige vers index.php. Elle permet aussi d’afficher à l’utilisateur les champs pour créer un nouveau compte.
* deconnecter.php : cette page n’affiche rien, mais enlève toutes les valeurs de la variable $\_SESSION et redirige l’utilisateur vers l’index.
* histoire.php : cette page affiche une histoire à l’utilisateur avec en plus de la moyenne de l’histoire et la moyenne de l’utilisateur, 4 listes déroulantes avec chacun d’eux les valeurs de 1-5 de pour chaque note pour noter l’histoire et si l’utilisateur est connecté un bouton pour rajouter l’histoire a ses favoris / enlever des ses favoris.
* Index.php : cette page affiche les histoires que le contrôleur lui envoyé trier soit par date de création (ordre décroissant) soit pas par note (ordre décroissant) et si l’utilisateur est connecté on affiche les histoires favorites.
* modifierHistoire.php : cette page permet d’afficher les champs pour modifier une histoire ou même créer une histoire.
* myCSS.css : CSS globale de l’application pour gérer l’affichage des images de l’histoire
* navbar.php : ce fichier est inclus dans toute les vus et permet gérer la barre de navigation et d’afficher différents liens dépendants de si utilisateur et connecter ou non.
* rechercher.php : cette page affiche le résultat d’une recherche avec toutes les histoires ayant le même titre que la recherche et toutes les histoires d’un utilisateur qui a le même nom que la recherche
* supprimer.php : permets d’afficher deux boutons oui / non pour supprimer une histoire

### Tales of the tavern/Controleur

Ce dossier contient les diffèrent contrôleur de chacune des vus en plus d’un contrôleur global.

* .htaccess : permets d’éviter à des utilisateurs de pouvoir rentrer dans ce dossier sur le site web.
* controleur.inc.php : Ceci est le contrôleur principal et donc toutes les requêtes qui demande des informations a la base de données sont ici de plus il contient les fonctions récurrentes qui sont utilisées sur plusieurs pages
* compte.inc.php : d’abord une requête SQL est faite avec l’email de l’utilisateur qui est stocké dans $\_SESSION [utilisateur] si aucun utilisateur existe alors on le redirige sur la page login.php. Sinon, on affiche le nom et l’email dans leurs champs respective. S’il y a des valeurs en POST et que le SHA256 du champs « ancien mot de passe » est égale au mot de passe stocker dans la base. Alors, on modifie le nom et l’email (en vérifiant que cet email n’est pas déjà utilisé), de plus, si les champs « nouveau mot de passe » et « confirmer nouveau mot de passe » ne sont pas nul et respecte la condition de 8 charactères de long minimum et contient au moins une lettre et un chiffre. Alors on modifie aussi le mot de passe de l’utilisateur. On appelle aussi la fonction AfficherHistoire() qui est dans le « controleur.inc.php » et en mettant la valeur true au paramètre « afficherBouton » cela permet d’afficher les boutons de modification(redirection vers modifierHistoire.php avec en GET l’id de l’histoire) et de suppression (redirection vers supprimer.php avec en GET l’id de l’histoire).
* connexion.inc.php : S’il y a des valeur en POST alors on vérifie que email que l’utilisateur nous a fournis existe bien dans notre base et ensuite on vérifie que le SHA256 de l’utilisateur est égale au mot de passe stocker dans la base si c’est le cas, on stock l’email dans $\_SESSION[utilisateur] sinon on affiche un message d’erreur.
* creerCompte.inc.php : S’il y a des valeurs en POST alors on vérifie que l’email que l’utilisateur nous a fournis n’existe pas dans notre base. Sinon, on affiche un message d’erreur. Ensuite, on vérifie que le mot de passe est égal au mot de passe de confirmation et que il fasse 8 charactère de long minimum et qu’il contient au moins une lettre et un chiffre. Si toutes ses conditions sont réunie alors on stock ces informations dans la base et comme pour la connexion on stock l’email dans $\_SESSION[utilisateur].
* histoire.inc.php : Si la variable ID en GET n’est pas égale a l’id d’une histoire dans la base de données alors on redirige l’utilisateur vers la page index.php. Sinon, on vérifie que $\_SESSION[utilisateur] existe et si c’est le cas on affiche le bouton ajouter/supprimer aux favoris si utilisateur a cette histoire en favoris ou non. Enfin si il y a des valeur en POST alors et que $\_POST[favoris] alors on insère les valeurs des énum dans la base de données or, $\_POST[favoris] existe c’est-à-dire que l’utilisateur veux rajouter au favoris l’histoire et donc on ne fais que rajouter l’histoire a ces favoris.
* Index.inc.php : Si la valeur de la variable ordre en GET n’est pas égal a moyenne ou n’existe pas, alors on récupère les histoires et les histoires favoris si l’utilisateur est connecté, trier par la date avec un ordre décroissant. Sinon on affiche les histoires, et les favoris, trier par moyenne aussi avec un ordre décroissant.
* modifierHistoire.php : si id en GET n’existe pas alors on affiche la vue avec aucune information stocker dans les champs et dès qu’on appuie sur le bouton ça ajoute une nouvelle histoire. Sinon, on vérifie que l’histoire existe bien et que c’est bien l’utilisateur qui a créé cette histoire (si ça n’est pas le cas, alors on le redirige vers. Mon compte.php) alors dès qu’il appuie sur le bouton on modifie son histoire et, s’il n’a pas rajouté d’image alors on laisse l’ancienne image.
* Rechercher.inc.php : Si en GET la variable recherche existe alors on demande au contrôleur de nous retourner toutes les histoires contenant se texte et si un utilisateur a se nom de retourner toutes ses histoires. Sinon on redirige l’utilisateur vers index.php
* Supprimer.inc.php : si id en GET n’existe pas ou que ce n’est pas cette utilisateur qui a créé cette histoire, alors on redirige l’utilisateur vers compte.php. Sinon, alors dès qu’il appuie sur le bouton oui on le redirige vers la page compte.php en supprimant les informations de l’hitoires. Si l’utilisateur appuie sur non alors on le redirige juste vers compte.php.

### Tales of the tavern/Model

* htaccess : permet d’éviter a des utilisateurs de pouvoir rentrer dans ce dossier sur le site web.
* connexionDB.php: ce fichier contient la fonction qui permet de se connecter à la base de données.
* constanteDB.php : ce fichier contient les constantes pour se connecter a la base de données username, password, **connexion,** server.
* fonctionDB.php : ce fichier contient toutes les que le contrôleur.inc.php va appeler pour envoyer au retourner des valeurs depuis la base

### Tales of the tavern/bootstrap-4.3.1-dist

Dossier contenant les fichiers nécessaires à Bootstrap.

### Tales of the tavern/Img

Ce dossier contient toutes les images.

## Analyse d’une fonctionnalité principale

# Test et plan de test

Tableau 2 : test des fonctionnalités

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Test | Résultat attendu | Résultat produit | Résultat |
| Fonctionnalité | | | | |
| F1 | Création compte + connexion | Un utilisateur peut créer un compte et en créer un nouveau | Il y a une page connexion.php qui permet de se connecter et une page creerCompte.php | OK |
| F2 | Gestion informations personnelles | Un utilisateur peut modifier ses informations | Un utilisateur peut changer ses informations dans la page compte.php | OK |
| F3 | Gestion histoire | Un utilisateur peut ajouter / supprimer / modifier une histoire | Un utilisateur peut ajouter / modifier / supprimer ses histoires dans la page compte.php | OK |
| F4 | Trie page accueil | Un utilisateur peut trier la page d'accueil par note ou par moyenne (ordre décroissant) | Un utilisateur peut trier la page d'acceuil avec 2 bouton | OK |
| F5 | Afficher favoris | Un utilisateur connecter à sa liste de favoris afficher sur l'accueil. | Si l'utilisateur est connecté alors les favoris s'affichent | OK |
| F6 | évaluer une histoire | Un utilisateur peut évaluer une histoire sur 4 critères | On peut noter une histoire sur 4 critères sur la page histoire.php | OK |
| F7 | Afficher moyenne utilisateur et histoire | Un utilisateur peut voir la moyenne d'une histoire et d'un auteur. | Un utilisateur peut voir la moyenne d'une histoire et la moyenne de l'auteur dans la page histoire.php | OK |
| F8 | Rechercher une histoire par le titre ou l'auteur | Un utilisateur peut rechercher une histoire par le titre ou l'auteur. | la barre de recherche qui est sur la barre de navigation permet de chercher une histoire par le titre et toutes les histoires d'un auteur | OK |

# Conclusion

## Objectifs

Je suis très content du travail que j’ai fourni et de l’état de actuel de mon projet. Il respecte

le cahier des charges et il correspond au projet que je souhaitais réaliser.

## Difficultés rencontrées

J’ai rencontré quelques difficultés lors de la création du script d’ajout d’avis pour les food

truck. Cependant, j’ai réussi à résoudre le problème assez vite. La plus grosse difficulté, je

l’ai eue lors de la modification de food trucks, car je n’avais aucune idée sur comment apporter cela au niveau de l’interface utilisateur pour la rendre le plus simple possible. J’ai

Demandé à mon professeur qui m’a proposé plusieurs solutions possibles.

## Apprentissages

//AJOUTE DE NOTE PAR LAUTEUR

## Remerciements

Je remercie le CFPT pour m’avoir permis de travailler dans de bonnes conditions. Je remercie

Plus particulièrement Mme Mota pour m’avoir suivi durant toute la durée du TPI et de

M’avoir donné des conseils pour la réalisation.

# Bibliographie

Manuel PHP : <https://www.php.net/manual/fr/index.php>

OpenClassroom : <https://openclassrooms.com/>

Stack Overflow : <https://stackoverflow.com/>

Github : <https://github.com/zabraf/TalesOfTheTavern>

# Annexes