****

**Sommaire**

**1. Introduction**

**Remerciements**

**Compétence du Référentiel**

**2. Résumé**

**L’Entreprise**

**Le Projet**

**3. Établissement du Cahier des Charges**

**De la Conception à la Création**

**L’interface Administrateurs**

**L’interface Utilisateurs**

**Gestion d’un Calendrier de Deadline Personnel**

**4. Etat de l’art**

**Architecture M.V.C : Définition et Fonctionnement**

**Les Avantages de se Modèle**

**5. Les Technologie Utilisées**

**Back-End**

**Front-End**

**6. La Décortication du Projet**

**L’arborescence de mes fichiers**

**Les Modèles**

**Les Contrôleurs**

**Les Vues**

**La Gestion des Erreurs et les Slugs pour le Référencement**

**Les Tests Réalisés**

**Un projet pour un autre projet**

**7. Les Erreurs et Bugs Rencontrés**

**Situations de travail ayant nécessité une recherche durant le projet**

**Conclusion**

**Introduction**

**Remerciements**

Avant de débuter, je tiens à exprimer ma gratitude envers toutes les personnes qui m'ont accompagné avant et pendant cette formation. Je souhaite remercier Pierre-Marie du D.A.Q de Luxeuil-les-Bains, qui m'a soutenu dans ma démarche et qui continue de le faire aujourd'hui.

Pendant la formation, je suis reconnaissant envers Allassane, le coach formateur de notre promotion, qui a été d'une aide inestimable pour me guider dans ma logique d'apprentissage. Je remercie également toute l'équipe d'Online Formapro pour leur compréhension et leur soutien.

Je suis particulièrement reconnaissant envers Jennyfer Coghe, ma référente de stage, qui m'a rapidement mis en confiance et m'a permis de mieux comprendre les besoins de l'entreprise. Je remercie également Isabelle et Laurent, les responsables du pôle culturel, qui m'ont fait découvrir les coulisses de leur travail et partagé leur expérience avec moi.

Enfin, je souhaite exprimer ma gratitude envers Online FormaPro pour leur compréhension et leur soutien, pour m'avoir donné l'opportunité de participer à cette formation, qui au départ me semblait impossible mais qui m'a permis de progresser et de m'ajouter de nouvelles compétences.

Je tiens à remercier tous ceux qui m'ont soutenu dans cette aventure et qui ont contribué à ma réussite. Votre soutien a été inestimable et je suis reconnaissant de l'avoir reçu.

Bonne Lecture,

Aurélien

**Compétences du Référentiel**

|  |  |  |
| --- | --- | --- |
| Activité Types | Compétence Professionnel |  |
| **Développer la partie Front-end d’une** | Maquetter une application |  |
| **application web ou web mobile en** | Réalisé une interface utilisateur web statique et adaptable |  |
| **intégrant les recommandations de** | Développer une interface utilisateur web dynamique |  |
| **sécurité** | Réalisé une interface utilisateur avec une solution de gestion de contenu |  |
| **Développer la partie Back-end d’une** | Créer une base de données |  |
| **application web ou web mobile en** | Développer les composants d’accès aux données |  |
| **intégrant les recommandations de** | Développer la partie Back-End d’une apllication web ou web mobile |  |
| **sécurité** | Elaborer et mettre en œuvre des composants dans une application de gestion de contenu ou e-commerce |  |

**Résumé**

**L’Entreprise**

J’ai effectué mon stage de fin de cursus au sein du Pôle Culturel de la Mairie de Luxeuil-les-bains.

Celui-ci est gérer par Laurent Wagner depuis 14 ans, il met en place les différents événements organisé par la ville et accueil les habitants pour la billetterie. Il gère les effets sonore et visuel de chaque spectacle en collaboration avec les équipes des différents évènements et assiste ensuite à la représentation. Il est assisté d’Isabelle depuis 3 ans qui tiens l’accueil avec lui et l’aide lors des spectacles.

**Le Projet**

Peut d’habitant de la ville connaisse le Pôle Culturel et ce qu’il organise. Il y a un dépliant répertoriant les spectacles mais ceux-ci ne sont disponibles qu’en Mairie, à l’Office du Tourisme, ou bien sûr au Pôle Culturel lui-même.

Pour aider Laurent et Isabelle afin de facilité la communication autour des événements de la ville et leur permettre de pouvoir administrer leur site sans connaissance fondamentale, j’ai donc proposé de créer un espace administrateur ou la quasi-totalité de l’application (Image d’accueil, textes, articles, couleur du site…) serait modifiable.

Leurs demandes après notre entretien : Que le site reprennent le style de la plaquette (image principale, période des événements, partenaires), que chaque articles aient un titre, une date, un prix selon les conditions du client (abonnée ou non, résident en ville ou non…), le synopsis du spectacle, un lien ou une vidéo si elle existe du teaser de l’événement, un fil d’actualité permettant à l’utilisateur de savoir quels sont les prochains spectacle et un formulaire permettant de se renseigner sur un prochain spectacle. Dans l’avenir et lorsque leur gestion leur permettra, ils aimeraient aussi pouvoir proposé un service de billetterie en ligne.

L’objectif étant bien sûr de proposé une interface intuitive pour l’utilisateur, facile à prendre en main et bien sûr moderne.

J'ai donc été amené à concevoir deux maquettes Figma pour l’application web. L'objectif était de proposer deux designs différents pour l’interface graphique, afin de leur permettre de choisir la meilleure option.

**Présentation des maquettes :**

La première maquette que j'ai proposée se concentrait sur une interface simple et épurée. Les éléments de l'interface étaient disposés de manière claire et intuitive, avec des couleurs sobres et des icônes épurées. L'accent était mis sur la lisibilité et l'ergonomie de l'interface, avec un souci du détail pour chaque élément de la page.



La seconde maquette que j'ai proposée se distinguait par un design plus dynamique et coloré. Les couleurs étaient plus vives et les icônes plus détaillées, avec une mise en page plus complexe pour donner une impression de mouvement et de fluidité. L'accent était mis sur l'expérience utilisateur, avec des éléments interactifs pour guider l'utilisateur à travers le site.



**Analyse comparative des maquettes :**

Après avoir présenté les deux maquettes, nous avons mené une analyse comparative pour déterminer laquelle serait la plus adaptée auproduit. Nous avons pris en compte plusieurs critères, tels que l'ergonomie, la lisibilité, la cohérence avec notre identité visuelle, l'expérience utilisateur, etc.

**Conclusion :**

En conclusion, après cette analyse comparative, nous avons décidé que la deuxième maquette était la plus adaptée à notre produit. En effet, elle répondait mieux à nos besoins en termes de lisibilité et d'ergonomie, tout en restant cohérente avec notre identité visuelle existante. La proposition de deux maquettes Figma a été une étape importante dans le processus de conception de notre produit. Cela m’a permis de tester différentes options et de prendre une décision éclairée sur la meilleure option pour l’nterface graphique et la suite du projet. Les maquettes Figma ont été un outil précieux pour communiquer ma vision à l'équipe de la mairie.

**Établissement du Cahier des Charges**

**De la Conception à la Création**

Plusieurs étapes entre en compte lors des créations d’un M.V.C (Modèle, Vue, Contrôleur). Les voici :

* **Analyse des besoins :**

Lors de mon entretien avec Jennyfer Coghe (secrétaire du Maire) et Hélène (Responsable de la communication), j’ai pu en apprendre plus sur leur besoin. Le pôle culturel de la ville ne disposait d’aucun outil de communication digitale, ils leurs faillaient quelque chose d’assez proche de la maquette.

* **Conceptions de la maquette :**

J’ai commencé à créer la maquette en utilisant l’outil de prototypage Figma. Je voulais un rendu ou l’image de la plaquette devienne la vitrine du site. Une image qui au départ était en format carré.

Pour adapter l’image au format écran, j’ai utilisé Adobe Photoshop pour retravailler la photo et l’agrandir.

J’ai ensuite voulu lui donner un peu vie donc j’ai détouré le personnage. Crée des calques avec les différentes parties de son corps que je voulais animer et j’ai utilisé le logiciel Adobe Animate.

* **Développement du modèle de données :**

Pour la base de données, j’ai utilisé le logiciel Workbench de MYSQL en me connectant à mon espace. J’ai donc créé les entités principales et les relations entre elles.

* **Développement du code :**

J’ai utilisé HTML, CSS et JavaScript pour le Front-End et PHP pour le Back-End. J’ai donc créé des modules séparé pour rendre le code plus logique et maintenable.

* **Test :**

Plusieurs test ont été pratiqué afin de vérifié que la version de mon application était une version stable et sans bug sur les fonctions importante de l’application :

* + L’ajout, l’édition où la suppression d’articles, l’ajout ou la suppression des images .
  + Si les erreurs 301, 404 étaient bien pris en compte.
  + La connexion à la base de données

**-Déploiement :**

Une fois que tous les tests auront été effectués avec succès, le site sera prêt à être déployé. Après un rendez-vous avec l’informaticien disposant des codes d’accès de l’hébergement, je configurerais celui-ci.

**L’interface Administrateurs**

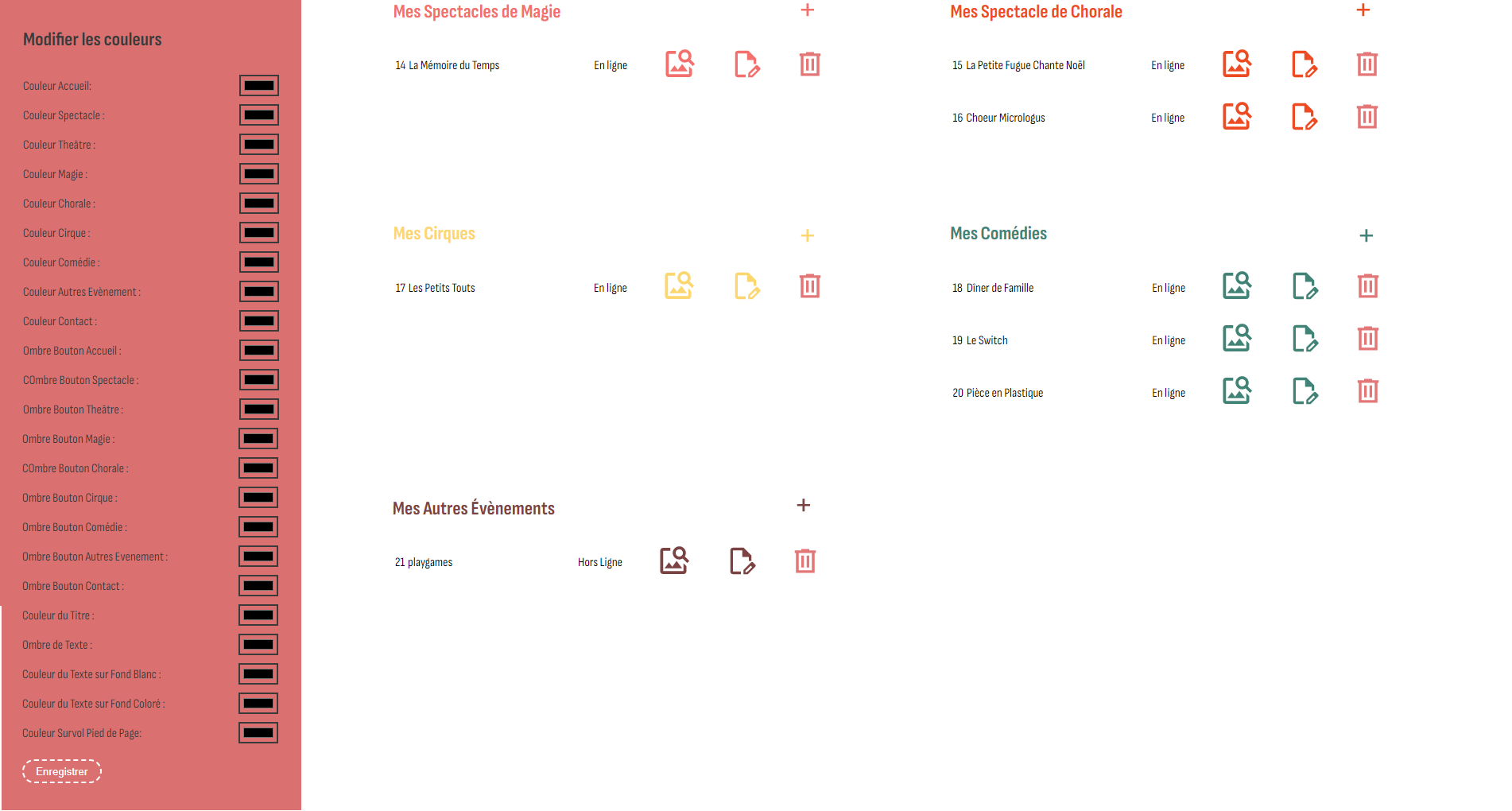
Pour facilité la compréhension, j’ai opté pour une interface épuré, qui permets à l’administrateur de s’y retrouvé entre les différentes composantes du système. Lors de la connexion à l’espace d’administration disponible de deux manières différentes :

* En cliquant sur ‘’Espace Administrateur’’ disponible via la page principale du site dans le Footer et seulement cette page. Ce lien ne s’affiche que sur une vue desktop, nous ne pouvons pas y accéder depuis un téléphone ou une tablette.
* Ou en entrant un mot clé dans le cas présent cockpit, modifiable si trop de personne essai d’y accéder.

L’administrateur pourra ainsi via ces deux solutions accéder à l’espace de connexion de l’administration. Une fois son identifiant et son mot de passe entrée (élément que nous aurons ensemble mis en place ensemble), il accèdera à l’espace administrateur.



Sur la gauche, la partie page avec l’édition et la suppression de chaque page du site, Au centre, chaque article avec l’ajout d’images pour chaque article, l’édition et la suppression. A côté de la nomenclature de chaque catégorie d’articles, la possibilité d’ajouter un article.



Lorsque l’administrateur scroll, celui-ci à accès à la modification des couleurs. Sous forme de input, il choisira lui-même comment il veut repenser le site et enregistrera ensuite le formulaire qui réécrira le code root.css. Cette interface me paraît simple, lisible et maximise le visuel. Il permet à l’administrateur d’avoir une gestion claire de son contenu.

**L’interface Utilisateurs**

Comme prévu lors de mon entretien avec le client, le site reflète le dépliant, j’ai adapté la chartre graphique au site pour que les gens habitués au format papier puissent se familiarisés assez vite avec l’interface. Sur la page d’accueil, une version étendu grâce à Adobe Photoshop et animé grâce à Adobe Animate permet d’avoir un premier visuel dynamique. Une barre de navigation sous forme de languette permet de rappeler le petit livret et elles ont une animation au survol qui permet à l’utilisateur une certaine interactivité.



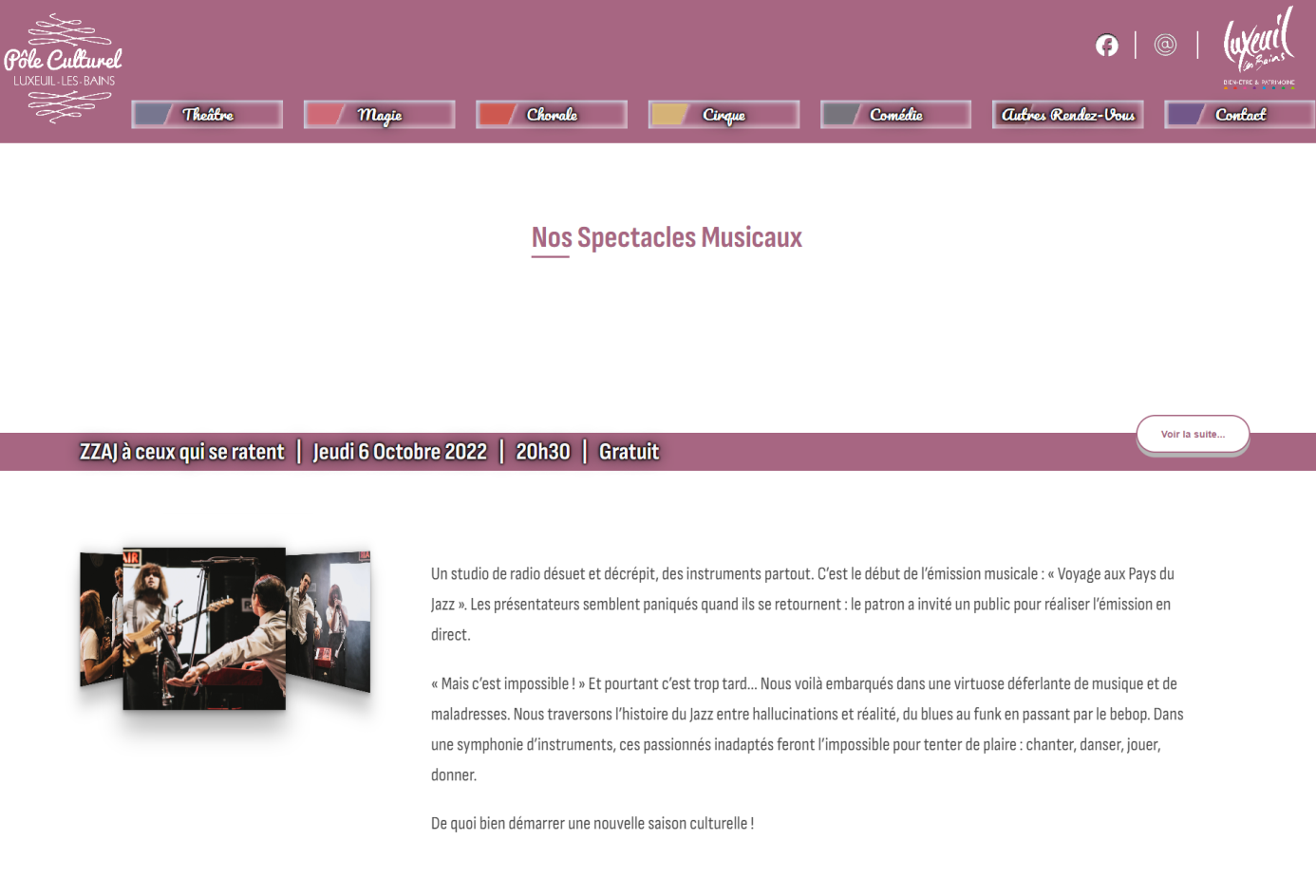
Lors du scroll, un fil d’actualité représentant les 3 prochains évènements apparaît. L’utilisateur dispose alors d’un bouton ‘’Réservez Ici’’ qui le redirigera vers la page contact et le formulaire de renseignement (en attendant une future billetterie en ligne).

Ensuite l’utilisateur accède à un lien ‘’Télécharger Ici’’, qui lui permettra de télécharger la plaquette des événements en version PDF, en dessous, un carrousel des différents partenaires financiers. Au survol de la souris, les logos qui étaient d’abord gris reprennent leur couleur initial et s’agrandissent.

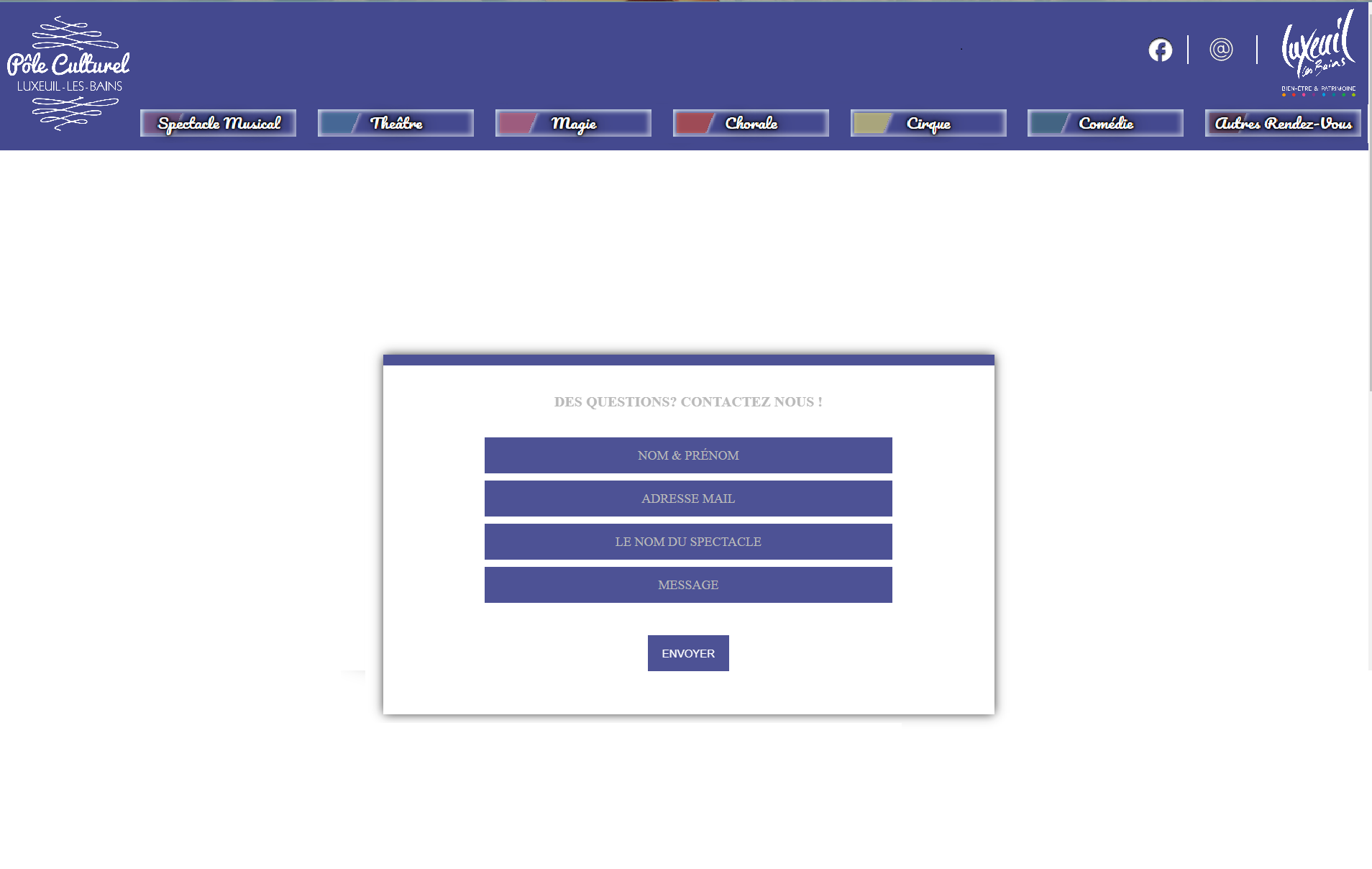
Et pour finir un pied de page où l’on retrouvera le lien Facebook du pôle culturel, les sites de la ville et de l’office du tourisme, les redirections vers les pages d’événements, l’adresse postale et le numéro de téléphone pour contacter le pôle culturel.

Lorsque l’utilisateur clique sur l’une des languettes d’événements chaque vue aura comme thème la couleur de l’événement.

Le logo Pôle Culturel sera quand à lui disponible en temps que lien afin de ramener l’utilisateur à la page d’accueil du site. Les différents articles se disposeront tous de façon à ne voir que le nom du spectacle, l’heure le lieu et le synopsis pour que lorsque l’utilisateur clique sur le bouton ‘’Voir la suite’’, il soit redirigé vers une autre page où les informations seront plus grosses et avec plus de contenu (dont le lien Youtube par exemple).



Une page contact sera bien sûr disponible, avec un formulaire renvoyant une méthode post et une action mai() permettant à l’utilisateur de demander quelques renseignements. Ceux-ci étant postés sur l’adresse mail, choisis par le Pôle Culturel.



**Gestion d’un Calendrier de Deadline Personnel**

Au commencement du projet et pour le mener à bien, je me suis imposé une deadline, pour chaque grandes étapes de mon projet afin de pouvoir le rendre dans le temps imparti :

* + Les maquettes : 2 Jours
  + Agrandissement de l’image et animations : 1 Jour
  + Création de la Base de Données : 1 Jour
  + Création de la Page d’Accueil Style + Responsive : 7 Jours
  + Création du Style + Responsive pour chaque Page Evénements : 7 Jours
  + Création d’une Logique M.V.C maintenable : 14 Jours
  + Correction des bugs et erreurs de logique : 7 Jours
  + Adaptation du projet au M.V.C : 7 Jours
  + Ajout contenu Base de Données (Contenu + Image) : 1 Jour
  + Création de l’Espace Administrateur + Style : 3 Jours
  + Ajout d’une fonction mail() + Style page contact : 2 Jours
  + Préparation au déploiement : 2 Jours
  + Déploiement : 1 Jour

Bien sûr c’était pour moi un prévisionnel, une fois sur le projet. Des choses m’ont demandé plus de temps et d’autre moins. Mais cela me laissait environ 1 semaine pour corriger ce qui ne fonctionnait pas, ajouter des fonctionnalités…

**Etat de l’art**

**Architecture M.V.C : Définition et Fonctionnement**

L'architecture Modèle-Vue-Contrôleur (MVC) est un modèle de conception logicielle utilisé pour développer des applications web. Il se compose de trois parties distinctes :

Le modèle (Model) : C'est la partie de l'application qui gère les données et les opérations liées à ces données. Il peut s'agir d'une base de données, d'un fichier texte, ou de tout autre support de stockage de données. Le modèle gère les opérations de création, de lecture, de mise à jour et de suppression (CRUD) des données.

La vue (View) : C'est la partie de l'application qui s'occupe de l'interface utilisateur. Elle permet d'afficher les données stockées dans le modèle de manière claire et conviviale. La vue est souvent constituée de pages web, d'éléments graphiques, de formulaires, etc.

Le contrôleur (Controller) : C'est la partie de l'application qui fait le lien entre la vue et le modèle. Il reçoit les entrées de l'utilisateur via la vue, traite ces entrées et utilise le modèle pour effectuer les opérations demandées. Le contrôleur peut également mettre à jour la vue pour refléter les changements effectués dans le modèle.

Le fonctionnement du MVC repose sur une séparation claire des responsabilités entre les trois parties. La vue est chargée d'afficher les données, le modèle de les stocker et le contrôleur de les manipuler. Cette séparation permet de faciliter la maintenance et l'évolution de l'application, car chaque partie peut être modifiée indépendamment des autres sans affecter leur fonctionnement global.

En somme, l'architecture MVC offre un cadre clair pour le développement d'applications web en séparant les différentes parties de l'application en couches distinctes. Cela facilite le développement, la maintenance et l'évolutivité de l'application.

**Les Avantages de se Modèle**

L’architecture Modèle-Vue-Contrôleur (MVC) présente de nombreux avantages pour le développement d'applications web :

Séparation des responsabilités : Le MVC permet de séparer clairement les responsabilités des différents composants de l'application. La vue est responsable de l'interface utilisateur, le modèle gère les données et le contrôleur sert de pont entre les deux. Cette séparation facilite la maintenance et l'évolutivité de l'application.

Réutilisabilité : La séparation des responsabilités facilite la réutilisation de code. Par exemple, le modèle peut être réutilisé dans plusieurs applications sans avoir à modifier la vue ou le contrôleur.

Développement plus rapide : Le MVC permet un développement plus rapide car chaque composant peut être développé indépendamment des autres. Cela permet aux développeurs de travailler simultanément sur différentes parties de l'application.

Facilité de test : Le MVC facilite les tests unitaires, car chaque composant peut être testé indépendamment des autres. Cela permet de détecter rapidement les erreurs et de les corriger plus facilement.

Meilleure évolutivité : Le MVC permet une meilleure évolutivité de l'application, car chaque composant peut être modifié indépendamment des autres. Cela permet de rajouter de nouvelles fonctionnalités ou de modifier l'existant sans affecter le reste de l'application.

En somme, l'architecture MVC est un modèle de conception logicielle efficace pour le développement d'applications web. Elle permet une séparation claire des responsabilités, une meilleure réutilisabilité, un développement plus rapide, une facilité de test et une meilleure évolutivité.

**Les Technologie Utilisées**

**Back-End  **

Pour le développement de mon back-end, j'ai choisi d'utiliser le langage de programmation PHP natif. C'est un langage de script côté serveur, conçu pour la création de sites web dynamiques. Pour m'assurer de la qualité de mon code, j'ai opté pour une fonction debug pour le test unitaire. Cela m'a permis de tester efficacement chaque unité de code et de m'assurer de son bon fonctionnement de manière isolée.

En ce qui concerne la base de données, j'ai utilisé MySQL, un système de gestion de base de données. Pour administrer et gérer la base de données, j'ai utilisé PHP MyAdmin. En ce qui concerne le serveur web, j'ai choisi d'utiliser Apache.

En utilisant cette combinaison de technologies, j'ai réussi à développer un backend efficace et fiable pour mon projet. Les tests unitaires effectués m'ont permis de garantir la qualité de mon code.

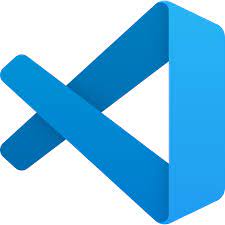
**Front-End** 

Pour la partie front-end de mon projet, j'ai utilisé les langages HTML, CSS et JavaScript en natif. J'ai choisi ces langages car ils sont largement utilisés pour le développement web et sont compatibles avec tous les navigateurs modernes.

Pour tester l'interface utilisateur de mon application web, j'ai utilisé les navigateurs les plus populaires tels que Chrome, Firefox et Edge. J'ai vérifié la compatibilité de l'application avec ces navigateurs pour m'assurer que les utilisateurs pourraient accéder à l'application sans problème.

Enfin, j'ai assuré la qualité de mon code front-end en effectuant des tests manuels et en utilisant les outils de développement des navigateurs pour identifier les problèmes et les corriger. Cela m'a permis de garantir que l'interface utilisateur de mon application web était intuitive, facile à utiliser et compatible avec différents navigateurs.

En utilisant ces langages, outils et navigateurs, j'ai réussi à développer une interface utilisateur web efficace pour mon projet, qui répond aux besoins de l'utilisateur et est compatible avec différents navigateurs.



J'ai utilisé Visual Studio Code comme outil de développement. Cet éditeur m’a offert une grande variété de fonctionnalités pour le développement web, telles que la coloration syntaxique, l'auto complétion et le débogage.



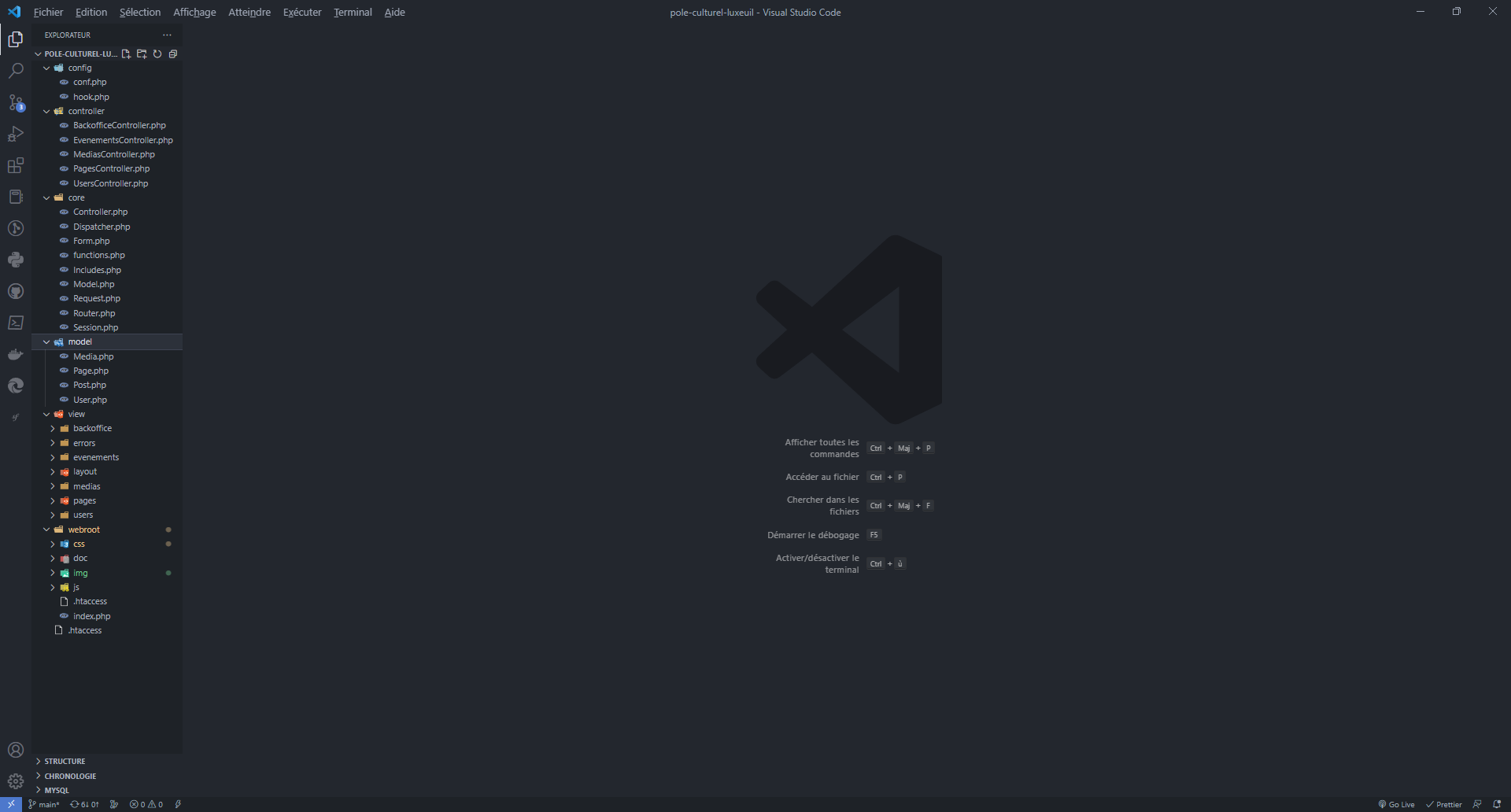
Pour la gestion de version de mon code, j'ai utilisé Git. Cela m'a permis de travailler sur différentes fonctionnalités de l'application simultanément, de fusionner mes modifications et de revenir en arrière en cas de besoin.



Pour héberger mon application web, j'ai choisi Ionos, un service d'hébergement web fiable et abordable.

**La Décortication du Projet**

**L’arborescence de mes Fichiers**

****

Le projet se compose de plusieurs dossiers et fichiers, organisés selon une architecture MVC (Modèle-Vue-Contrôleur), ce qui permet de séparer les différentes parties de l'application et de faciliter leur gestion et leur maintenance.

**Dossier Config :**

Le dossier config contient le fichier conf.php, qui permet de stocker les variables de connexion à la base de données ainsi que les préfixes de connexion.

**Dossier Controller :**

Le dossier contrôleur contient les fichiers du contrôleur :

BackofficeController.php : contient les fonctions liées à la partie administration de l'application.

EvenementsController.php : contient les fonctions liées aux événements.

MediasController.php : contient les fonctions liées à l'administration des images postées dans les articles.

PagesController.php : contient les fonctions liées aux pages.

UsersController.php : contient les fonctions liées à la gestion de la connexion des utilisateurs.

**Dossier Core :**

Le dossier core contient les fichiers centraux de l'application :

Controller.php : contient les fonctions construct, render, set, loadModel, e404, request, redirect.

Dispatcher.php : contient les fonctions construct, error, loadController.

Functions.php : contient la fonction debug.

Includes.php : contient les inclusions.

Model.php : contient les fonctions construct, find, findCount, delete, save.

Request.php : contient la fonction construct.

Router.php : contient les fonctions prefix, parse, connect, url, webroot.

Session.php : contient les fonctions construct, setFlash, flash, write, read, isLogged.

Dossier model

Le dossier model contient les fichiers de modèles :

Media.php : représente les images postées dans les articles.

Page.php : représente les pages du site. Contient la fonction validates pour valider les données.

Post.php : représente les articles du site. Contient la fonction validates pour valider les données.

User.php : représente les utilisateurs du site.

**Dossier View :**

Le dossier view contient les fichiers de la vues :

Dossier backoffice : contient les fichiers de vues liés à la partie administration de l'application.

Dossier errors : contient le fichier e404.php, qui est affiché lorsque l'utilisateur accède à une page inexistante.

Dossier evenements : contient les fichiers de vues liés aux événements.

Dossier layout : contient les fichiers de mise en page par défaut de l'application.

Dossier medias : contient les fichiers de vues liés à l'administration des images postées dans les articles.

Dossier pages : contient les fichiers de vues liés aux pages du site.

Dossier users : contient les fichiers de vues liés à la gestion de la connexion des utilisateurs.

**Dossier Webroot :**

Le dossier webroot contient les fichiers publics de l'application :

Dossier css : contient les fichiers CSS liés au style des vues.

Dossier js : contient les fichiers JavaScript de l'application.

Dossier doc : contient les fichiers de documentation du projet.

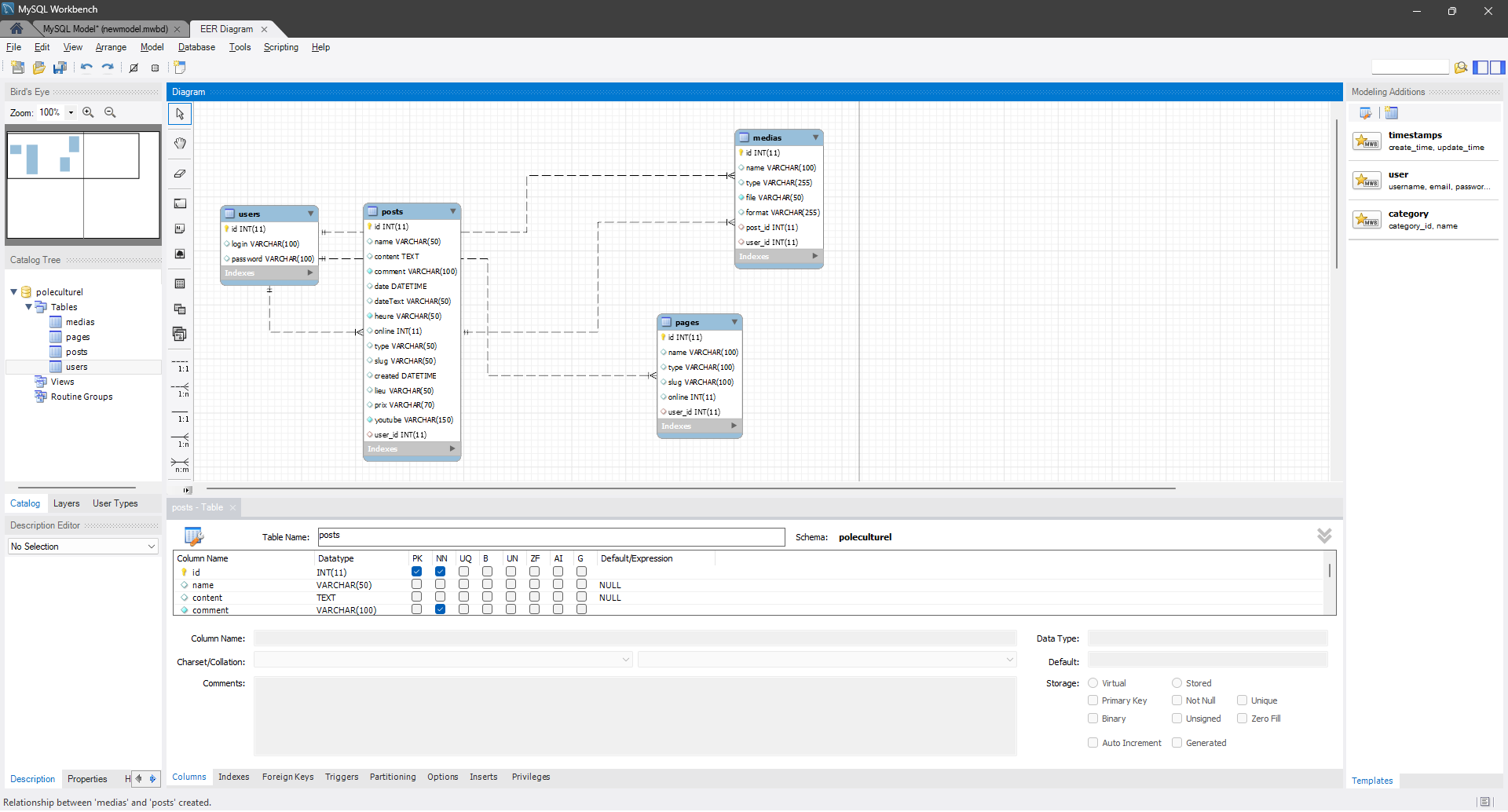
Dossier img : contient les images utilisées par l'application.

.htaccess : fichier de configuration d'Apache pour les règles de redirections.

index.php : point d'entrée de l'application.

**Les Modèles**

Le modèle de l'application est responsable de l'accès et de la gestion des données. Il contient les classes pour interagir avec la base de données, valider et manipuler les données, et récupérer les informations pour les contrôleurs.



Dans ce projet, le modèle se compose de quatre classes principales :

La classe User : Cette classe est responsable de la gestion des utilisateurs. Elle contient les propriétés et les méthodes nécessaires pour créer, lire, mettre à jour et supprimer des utilisateurs dans la base de données. Elle est également responsable de la validation des données pour s'assurer que les utilisateurs sont correctement authentifiés.

Création de la table Users : Permet de récupérer les informations de chaque administrateur contenant l’ID, le login et le mot de passe avec une clé de cryptage SHA1

La classe Post : Cette classe est responsable de la gestion des articles. Elle contient les propriétés et les méthodes nécessaires pour créer, lire, mettre à jour et supprimer des articles dans la base de données. Elle est également responsable de la validation des données pour s'assurer que les articles sont correctement formatés

Création de la table Posts : Permet de stocker chaque article avec l’ID, le titre, le contenu, la date (en chaîne de caractère), la date (avec la méthode dateTime pour le file d’actualité), l’heure, le type (Spectacle Musical, Théâtre….), le slug (qui me permet de récupérer le nom du spectacle dans l’url), le lieu, le prix, le lien youtube, online qui détermine si l’article est en ligne ou non et je l’ai relié à l’ID de la table Users pour identifié la personne ayant fais les modifications.

La classe Page : Cette classe est responsable de la gestion des pages. Elle contient les propriétés et les méthodes nécessaires pour créer, lire, mettre à jour et supprimer des pages dans la base de données. Elle est également responsable de la validation des données pour s'assurer que les pages sont correctement formatées.

Création de la table Pages : Permets de gérer les pages de mon application en conservant le titre, le type, le slug, le fait qu’il soit en ligne ou non, et relié à l’ID de la table User

La classe Media : Cette classe est responsable de la gestion des médias associés aux articles. Elle contient les propriétés et les méthodes nécessaires pour créer, lire, mettre à jour et supprimer des médias dans la base de données. Elle est également responsable de la validation des données pour s'assurer que les médias sont correctement associés aux articles.

Création de la table Medias : Permet d’associer chaque image à chaque article avec un ID, le titre, le type, le format, j’ai relié chaque image à chaque ID de la table Posts et ID de la table Users.

Chacune de ces classes est basée sur une classe de base Model qui fournit les méthodes de base pour interagir avec la base de données. Les classes de modèle utilisent également la bibliothèque de validation pour valider les données avant de les enregistrer dans la base de données.

Enfin, les classes de modèle sont utilisées par les contrôleurs pour récupérer les données nécessaires pour générer les vues. Les contrôleurs appellent les méthodes des classes de modèle pour récupérer les données, puis transmettent ces données aux vues pour les afficher.

**Les Contrôleurs**

Le contrôleur contient les différents fichiers pour chaque fonctionnalité de l'application.

Chaque fichier de contrôleur contient des fonctions correspondantes aux différentes actions que l'utilisateur peut effectuer sur l'application. Par exemple, le fichier UserController.php contient les fonctions "login" pour gérer la connexion, la déconnexion et l'inscription des utilisateurs.

Chaque fonction est responsable de récupérer les données nécessaires auprès du modèle, de les traiter et de les renvoyer à la vue pour affichage. Le contrôleur est également chargé de gérer les erreurs et les exceptions qui peuvent survenir lors de l'exécution des fonctions.

Le contrôleur est également responsable de la gestion des routes de l'application. Il utilise le Router de l'application pour mapper les URL aux fonctions correspondantes du contrôleur. Cela permet à l'application de diriger correctement les utilisateurs vers la bonne fonctionnalité en fonction de l'URL demandée.

En conclusion, le contrôleur de l'application MVC est un élément clé de l'architecture de l'application. Il permet de gérer la logique de l'application, de traiter les demandes de l'utilisateur et de renvoyer les résultats à la vue. Les différents fichiers de contrôleurs permettent de séparer les fonctionnalités de l'application en fonction de leur domaine et de faciliter leur maintenance.

**Les Vues**

Le dossier view contient toutes les vues de l'application. Chaque vue est organisée en fonction du type de page qu'elle représente, tel que la page d'accueil, la page de contact, les pages d'événements.

Les pages de l'application sont créées en utilisant des fichiers de mise en page HTML avec des blocs de code PHP intégrés pour afficher les données dynamiques de l'application. Les pages peuvent inclure des éléments HTML tels que des formulaires, des tableaux, des listes et des images pour présenter les données de manière conviviale.

Les fichiers de mise en page sont organisés dans le dossier Layout et sont appelés par les contrôleurs pour ajouter un en-tête et un pied de page uniforme à chaque page. Les fichiers de mise en page peuvent également inclure des scripts JavaScript et des feuilles de style CSS pour la mise en forme et l'interactivité des éléments de la page.

Les vues sont organisées en fonction des contrôleurs qui les utilisent. Par exemple, le contrôleur des événements aura ses propres vues pour afficher un événement individuel, la liste des événements, la création et la modification d’événements.

Les vues sont créées en utilisant le langage de Template PHP pour faciliter la réutilisation et la maintenance du code. Les variables PHP sont utilisés pour passer les données dynamiques à partir des contrôleurs aux vues.

**La Gestion des Erreurs et les Slug pour le Référencement**

Je sais que la gestion des erreurs 404 et 301 est essentielle pour assurer une expérience utilisateur optimale et pour améliorer le référencement de mon site. Une erreur 404 se produit lorsque le serveur ne peut pas trouver la page demandée par l'utilisateur. Pour éviter cela, j'ai mis en place des pages d'erreur 404 bien conçues pour aider à maintenir l'engagement des utilisateurs et les encourager à explorer d’avantage mon site.

D'autre part, j'utilise également des redirections 301 pour aider les utilisateurs à trouver rapidement et facilement les informations qu'ils recherchent. Les redirections 301 peuvent également aider à transférer le jus de lien et les signaux de référencement de la page d'origine vers la nouvelle page, ce qui peut améliorer le référencement de mon site Web.

En ce qui concerne les slug, j'ai veillé à ce qu'ils soient descriptifs, courts et pertinents pour le contenu de chaque page. Cela aide à améliorer la convivialité de mon site Web et facilite la recherche et la découverte de mon contenu par les utilisateurs et les moteurs de recherche.

Enfin, j'ai également pris en compte l'importance des noms d'image pour le référencement. Les noms d'image doivent être descriptifs et pertinents pour le contenu de l'image. Ils aident les moteurs de recherche à comprendre le contenu de l'image et peuvent également améliorer la recherche d'images sur mon site Web. Par conséquent, j'ai choisi des noms d'image courts, concis et utilisant des mots-clés pertinents pour améliorer leur référencement.

En résumé, j'ai pris en compte l'importance de la gestion des erreurs 404 et 301, les slug et les noms d'image pour améliorer la convivialité et le référencement de mon site. En utilisant ces bonnes pratiques, je peux garantir que mon site est facilement accessible et facile à naviguer pour les utilisateurs et les moteurs de recherche.

**Les Tests Réalisés**

Tests unitaires : ces tests visent à vérifier que chaque composant (ou unité) de mon application fonctionnaient correctement de manière isolée.

Tests fonctionnels : ces tests visent à vérifier que l'application répond correctement aux exigences fonctionnelles et aux besoins des utilisateurs.

**Un Projet pour un Autre Projet**

Je sais que l'architecture MVC est largement utilisée dans les projets web, tels que les sites de E-commerce, les réseaux sociaux et les applications d'entreprise. Le dossier ‘’core’’ de mon projet, me permettra de conservé ce type d’architecture pour d’autres application du même type. Avec une plus grande expérience, je pourrais bien sûr y améliorer ces fonctionnalités.

Cependant, il est important de noter que la compatibilité entre les projets peut être affectée par des différences dans les versions de langage de programmation et les environnements d'exécution. Par conséquent, il est important de s'assurer que tous les projets impliqués utilisent les mêmes versions de logiciels.

Enfin, la compatibilité entre les projets peut être améliorée en utilisant des normes et des conventions communes pour les noms de fichiers, les paramètres et les variables. Cela facilite la communication et la compréhension entre les différents membres de l'équipe de développement.



**Les Erreurs et Bugs Rencontrés**

**Situations de travail ayant nécessité une recherche durant le projet**

Plusieurs problèmes ont eu raison de ma patience à toute épreuve :

* + - Les caractères manquant dans les expressions régulière (le + ou le \*)

qui détermine le nombre de fois que les caractères précédents apparaissent.

* + - Les caractères manquant dans les expressions régulière (le ^ ou le $)

qui détermine ce par quoi doit commencer par la chaîne de caractère qui suit ^ et doit se terminer par l’expression qui précède $

* + - Les erreurs de concaténation avec les commande SQL qui ramène une

erreur SQLSTATE[….].

* + - Le routing, par exemple pour ajouter des éléments du webroot et bien formater l’url de redirection.

Mais bien qu’impitoyable, toutes ces erreurs en prenant le temps, j’ai réussi à les traiter et cela à grandement renforcer mon expérience.

**Conclusions**

En conclusion, ce stage m'a permis de développer un projet utilisant l'architecture MVC, qui a constitué une expérience très enrichissante. Tout d'abord, j'ai pu apprendre en profondeur le fonctionnement de ce modèle, ses avantages et ses limitations, ainsi que les bonnes pratiques associées à son implémentation. J'ai également pu acquérir une expérience significative dans le développement web, en travaillant avec des technologies telles que PHP, MySQL, HTML, CSS et JavaScript.

En ce qui concerne la réalisation du projet, j'ai mis en place un système de gestion de contenu efficace, qui répond aux besoins de l'utilisateur tout en respectant les exigences de sécurité et de performance. J'ai également consacré une grande partie de mon temps à la mise en place de tests automatisés pour le back-end, qui ont permis de garantir la qualité du code et d'assurer une maintenance efficace à long terme.

En somme, ce stage m'a permis d'acquérir des compétences précieuses dans le domaine du développement web, ainsi que dans l'utilisation de l'architecture MVC. Je suis reconnaissant(e) envers mon tuteur pour son encadrement et ses conseils tout au long de cette expérience, ainsi qu'envers l'entreprise pour m'avoir offert cette opportunité de stage. Je suis désormais prêt(e) à mettre en pratique les compétences acquises dans de nouveaux projets, en continuant à apprendre et à me développer professionnellement.