

# *les7mainsmagiques*

21/11/2023

MICHOT Aglaé  
les7mainsmagiques  
56 rue Veron  
94140 Alfortville

## *Remerciements :*

*Je tiens à exprimer ma profonde gratitude à l'AFPA Le Havre pour son soutien exceptionnel tout au long de cette formation. Un immense merci à ma famille, en particulier à mon conjoint, pour leur encouragement constant. Un merci tout spécial à ma marraine pour sa bienveillance. Enfin, je tiens à adresser mes plus sincères remerciements à mon professeur, M. Anousone Mounivongs, dont l'expertise et l'engagement ont grandement contribué à l'enrichissement de mes connaissances et à la réussite de mon parcours. Sa guidance éclairée et son dévouement ont été des sources d'inspiration, m'incitant à atteindre mes objectifs avec détermination et assurance.*

## **Vue d'ensemble**

Dans le cadre de mon stage de deux mois et demi , je devais créer un site avec une bonne capacité de stockage afin qu'il puisse contenir tous les événements de l'association.

La conception du site web repose sur la création d'une plateforme efficace, répondant aux besoins spécifiques de l'organisation tout en offrant une expérience utilisateur optimale. Le processus débute par une analyse approfondie des objectifs de l'association, de son public cible et des fonctionnalités requises. En concertation avec la présidente de l'association, j'ai élaboré une architecture de site intuitive, mettant en avant les informations cruciales telles que les activités et les événements .

Le design, pensé pour refléter l'identité visuelle de l'association, a été développé avec une attention particulière à l'accessibilité et à la convivialité. La navigation a été optimisée pour garantir une expérience utilisateur fluide, favorisant l'interaction et la participation. La mise en œuvre de technologies web actuelles assure la compatibilité avec divers dispositifs, offrant une accessibilité accrue.

Des fonctionnalités de gestion de contenu ont été intégrées pour permettre à l'administrateur de mettre à jour facilement et régulièrement le contenu du site.

En conclusion, la conception du site pour les7mainsmagiques repose sur une approche holistique, alliant une compréhension approfondie des besoins de l'organisation, un design réfléchi, une navigation intuitive, une sécurité renforcée, et une adaptabilité aux différentes plateformes, le tout dans le but de renforcer la présence en ligne et d'encourager l'engagement de la communauté.

## **Objectifs**

1. Créer un site ayant les anciens et nouveaux événements
2. Créer un site avec une bonne capacité de stockage

## **Caractéristiques**

Le site web de l'association a été conçu avec des fonctionnalités robustes pour la gestion des événements, offrant une expérience interactive et dynamique. L'administrateur peut ajouter de nouveaux événements via une interface conviviale afin de remplir le site. La fonction "Ajouter un événement" permet de saisir les détails pertinents tels que le titre, la date, le lieu et une possible description.

Une section dédiée à la visualisation des événements assure une présentation claire et ordonnée de toutes les activités planifiées. Chaque événement est présenté de manière attrayante, mettant en avant les détails clés et les images associées.

Chaque visiteur peut s'inscrire à l'événement à condition de rentrer son adresse email et l'administrateur pourra supprimer son inscription ou ajouter la participation d'adhérent à un événement.

La fonction de modification offre aux administrateurs la possibilité de mettre à jour les informations des événements en temps réel. Que ce soit pour ajuster la date, modifier le lieu ou actualiser la description, cette fonctionnalité assure une gestion agile et réactive du contenu.

La suppression d'événements est également gérée de manière intuitive, permettant à administrateur de retirer rapidement des activités obsolètes ou annulées. Une confirmation est généralement requise pour éviter toute suppression accidentelle.

Enfin, la possibilité d'ajouter des photos aux événements enrichit l'expérience visuelle des utilisateurs. Les images peuvent être téléchargées à tout moment : lors de la création, de la modification d'un événement ou au moment de l'affichage, renforçant ainsi l'attractivité et l'engagement autour des activités de l'association. Ces fonctionnalités combinées offrent un outil complet de gestion des événements, permettant à l'association de maintenir un calendrier dynamique, informatif et visuellement attrayant sur son site web.

## Grandes étapes

- I. Cahier des charges
- II. Créer une maquette
- III. Créer une interface statique puis dynamique et adaptable
- IV. Créer une base de donnée
- V. Composants d'accès à la base de donnée
- VI. Création de l'espace administrateur
- VII. Gestion complète des événements avec sécurisation des données
- VIII. Gestion de l'ajout de photos aux événements

## Table des matières :

<i>I. Cahier des charges</i>	page 6
Descriptif de la demande Architecture du site Spécificités fonctionnelles	page 6

Ajout d'événement ou de photos Affichage des événements	page 7
Affichage des photos Modifications ou suppressions des événements ou des photos Inscription à un événement Spécifications techniques (langages, librairie et outils)	page 8
Sécurité des systèmes d'information recommandation et bonne pratique	page 10
<b>II. Compétences du référentiel couverte</b>	page 12
Tableau récapitulatif	page 12
Maquette de l'application	page 13
Réalisation d'une interface utilisateur web statique et adaptable	page 14
Développer une interface utilisateur web dynamique	page 16
Développer une interface utilisateur avec une solution de gestion de contenu	page 16
Création d'une base de données	page 18
Développer des composants d'accès aux données	page 22
Développer la partie Back-End d'une application web ou web mobile	page 23
<b>III. Réalisation</b>	page 27
Création d'un événement	page 27
Afficher un événement	page 30
Modifier un événement	page 34
Supprimer un événement	page 42
Anglais de rigueur : une communauté importante orientée vers le partage	page 45
Conclusion	page 47

# I. Cahier des charges

## Descriptif de la demande :

Développer un site web pour les7mainsmagiques afin de partager les informations clés sur ses missions et activités, et faciliter l'interaction avec le public.

Le site devra inclure une section dédiée aux événements, permettant l'ajout, la modification, et la suppression d'événements, ainsi que l'ajout d'images pour une expérience visuelle enrichie. La conception devra être intuitive, sécurisée, et adaptable à différents dispositifs.

## Architecture du site :

Le site correspondra à certains critères comme l'adaptabilité sur tout écran et sur tout type de navigateur ; les langages utilisés seront principalement HTML, CSS, Javascript et PHP ; l'usage d'une base de données MySQL servira de stockage pour les données ; le recours de la librairie bootstrap sera occasionnelle .

Le site sera développé sur un serveur LAMP et adoptera l'architecture Modèle-Vue-Contrôleur.

## Spécificités fonctionnelles :

Un événement est composé de :

- Un titre
- Un type
- Un lieu
- Une ou deux dates
- Une description (qui reste facultative)
- Un poster

Un type est composé de :

- Un nom

Un album est composé de :

- Un nom
- Un événement

Une photo est composé de :

- Un nom
- Un album

Un administrateur est composé de :

- Une adresse email (login)

- Un mot de passe

Un participant est composé de :

- Une adresse email
- Un événement

## Ajout d'événement ou de photos :

Le site permet à l'administrateur d'ajouter un événement à la base de données depuis la page programmation ou la page galerie .

Il pourra ensuite choisir d'ajouter des photos à l'événement qui seront téléchargées dans un dossier au nom de l'événement afin de ne pas avoir de problème de stockage dans la base de données .

## Affichage des événements :

Si l'événement est de type exposition et que la date de fin est antérieure à la date du jour il sera affiché dans la galerie à condition qu'il possède des photos . Sinon seul l'administrateur le verra affiché dans la galerie afin qu'il puisse soit ajouter des photos soit le supprimer .

Il en va de même pour le type sortie ou assemblée générale si leur date est antérieure à la date du jour . Cependant si leur date est ultérieure à la date du jour ils seront tous affichés sur la page programmation .

## Affichage des photos :

L'affichage des photos est disponible sur la page qui montre les détails de l'événement .

## Modifications ou suppressions des événements ou des photos :

L'administrateur peut à tout moment modifier, supprimer un événement ou ses photos depuis la page programmation ou galerie .

## Inscription à un événement :

Un visiteur peut à tout moment s'inscrire aux événements disponibles : sortie ou assemblée générale (à la demande de l'administrateur). L'administrateur peut lui aussi modifier la liste des participants en en supprimant ou en ajoutant des adhérents . Si le participant existe déjà, un message s'affiche afin de le signaler.

## Spécifications techniques (langages, librairie et outils) :

<u>Appellation</u>	<u>Description</u>
<b>HTML 5</b>	HTML5 est la dernière version du langage de balisage utilisé pour structurer le contenu des pages web, introduisant de nouvelles balises sémantiques, des fonctionnalités multimédias intégrées, et des améliorations significatives pour une conception web moderne et accessible.
<b>CSS 3</b>	Le CSS3 (Cascading Style Sheets 3) est la dernière version du langage de feuilles de style en cascade utilisé pour définir la présentation et la mise en forme des documents HTML et XML sur le web, introduisant des fonctionnalités avancées telles que les transitions, les animations, les ombres, et les grilles.
<b>Javascript</b>	JavaScript est un langage de programmation de haut niveau, interprété, principalement utilisé pour rendre les pages web interactives en permettant la manipulation dynamique du contenu et la gestion des événements utilisateur côté client.
<b>PHP 7</b>	PHP 7 est une version majeure du langage de programmation côté serveur, caractérisée par des améliorations significatives de performances, une gestion des erreurs renforcée, l'introduction du type de retour et la prise en charge accrue de la programmation orientée objet.
<b>MySQL</b>	MySQL est un système de gestion de base de données relationnelle open source, utilisant le langage de requête SQL (Structured Query Language) pour stocker, interroger et manipuler les données de manière efficace et structurée.
<b>Bootstrap</b>	Bootstrap est une librairie open-source de développement web, facilitant la création d'interfaces responsives et esthétiques grâce à ses composants prêts à l'emploi et son système de grille flexible.
<b>Visual Studio Code</b>	Visual Studio Code (VSCode) est un éditeur de code source léger, open-source et multiplateforme, développé par Microsoft. Il offre une interface utilisateur intuitive, des fonctionnalités d'édition avancées et une vaste prise en charge de langages de programmation, tout en permettant l'intégration de nombreuses extensions pour personnaliser l'environnement de développement.
<b>Git</b>	Git est un système de gestion de versions décentralisé, largement utilisé dans le développement de logiciels, permettant de suivre les modifications du code source, de collaborer efficacement entre plusieurs contributeurs, et de revenir à des versions antérieures du projet si nécessaire.

<b>Figma</b>	Figma est une plateforme de conception collaborative basée sur le cloud, offrant des outils de prototypage, de conception d'interfaces utilisateur et de collaboration en temps réel, permettant aux équipes de concevoir et de partager des projets de manière transparente.
<b>PhpMyAdmin</b>	PhpMyAdmin est une interface web open source permettant la gestion graphique des bases de données MySQL, offrant aux utilisateurs la possibilité de créer, modifier et interroger des bases de données de manière conviviale.
<b>MySQL Workbench</b>	MySQL Workbench est un outil de modélisation de bases de données et d'administration visuelle, facilitant la conception, la gestion et la maintenance des bases de données MySQL grâce à une interface graphique complète et des fonctionnalités avancées.
<b>JMerise</b>	JMerise est un outil de modélisation de bases de données relationnelles qui offre une interface graphique intuitive pour concevoir, visualiser et générer des schémas de bases de données. Il facilite la création de modèles entité-relation et la génération de scripts SQL correspondants.

## Sécurité des systèmes d'information : recommandations et bonnes pratiques :

La sécurité d'un site web est cruciale pour protéger les données sensibles, garantir la disponibilité du service et prévenir les attaques. Voici quelques pratiques de sécurité essentielles que j'ai utilisé pour le site web les7mainsmagiques:

1. Chiffrement HTTPS : J'utilise le protocole HTTPS pour chiffrer les communications entre le navigateur des utilisateurs et le serveur, assurant ainsi la confidentialité des données transitant sur le site.
2. Gestion des accès et des identités : Je met en œuvre une gestion stricte des droits d'accès pour limiter l'usage non autorisé des fonctionnalités du site. J'utilise des mécanismes d'authentification robustes, comme la double authentification.
3. Mises à jour régulières : Je garde le logiciel, les frameworks, les plugins et les systèmes d'exploitation à jour pour remédier aux vulnérabilités connues.
4. Protection contre les injections SQL : J'applique des filtres et des validations pour prévenir les attaques par injection SQL, une technique courante pour exploiter les failles de sécurité dans les bases de données.
5. Protection contre les attaques par force brute : Je met en place des mécanismes de blocage après un certain nombre de tentatives infructueuses de connexion pour prévenir les attaques par force brute.

6. Sécurisation des téléchargements de fichiers : Le site permettant le téléchargement de fichiers, je m'assure que ces fichiers sont sécurisés et je les scans régulièrement pour détecter tout contenu malveillant.
7. Protection contre les attaques par déni de service (DDoS) : J'utilise des services de mitigation DDoS pour prévenir ou atténuer les attaques qui pourraient entraîner une surcharge du site.
8. Sauvegardes régulières : J'effectue des sauvegardes régulières des données du site et je m'assure qu'elles peuvent être restaurées rapidement en cas de besoin.

Ces recommandations proviennent de sources telles que l'OWASP, le NIST et d'autres organismes de sécurité, et elles peuvent évoluer avec le temps. Elles seront appliquées tout au long du projet afin de renforcer la protection du site web contre une variété de menaces potentielles.

## II. Compétences du référentiel couvertes

Tableau récapitulatif :

N° fiche AT	Activités types	N° fiche CP	Compétences professionnelles
1	Développer la partie front-end d'une application web ou web mobile en intégrant les recommandations de sécurité	1	Maquetter une application
		2	Réaliser une interface utilisateur web statique et adaptable
		3	Développer une interface utilisateur web dynamique
		4	Réaliser une interface utilisateur avec une solution de gestion de contenu ou e-commerce
2	Développer la partie back-end d'une application web ou web mobile en intégrant les recommandations de sécurité	5	Créer une base de données
		6	Développer les composants d'accès aux données
		7	Développer la partie back-end d'une application web ou web mobile
		8	Elaborer et mettre en oeuvre des composants dans une application de gestion de contenu ou e-commerce

Légendes :

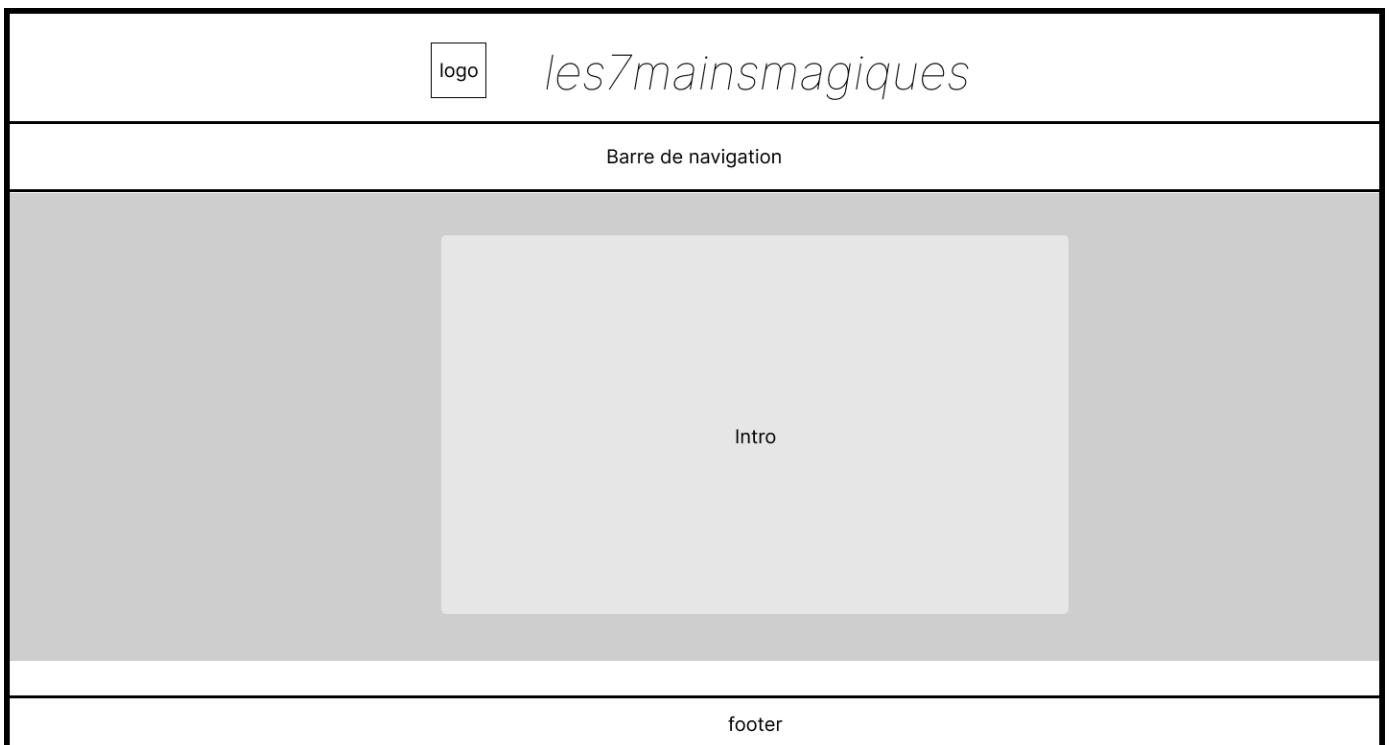
Compétences couvertes
Compétences non couvertes

# Maquette de l'application

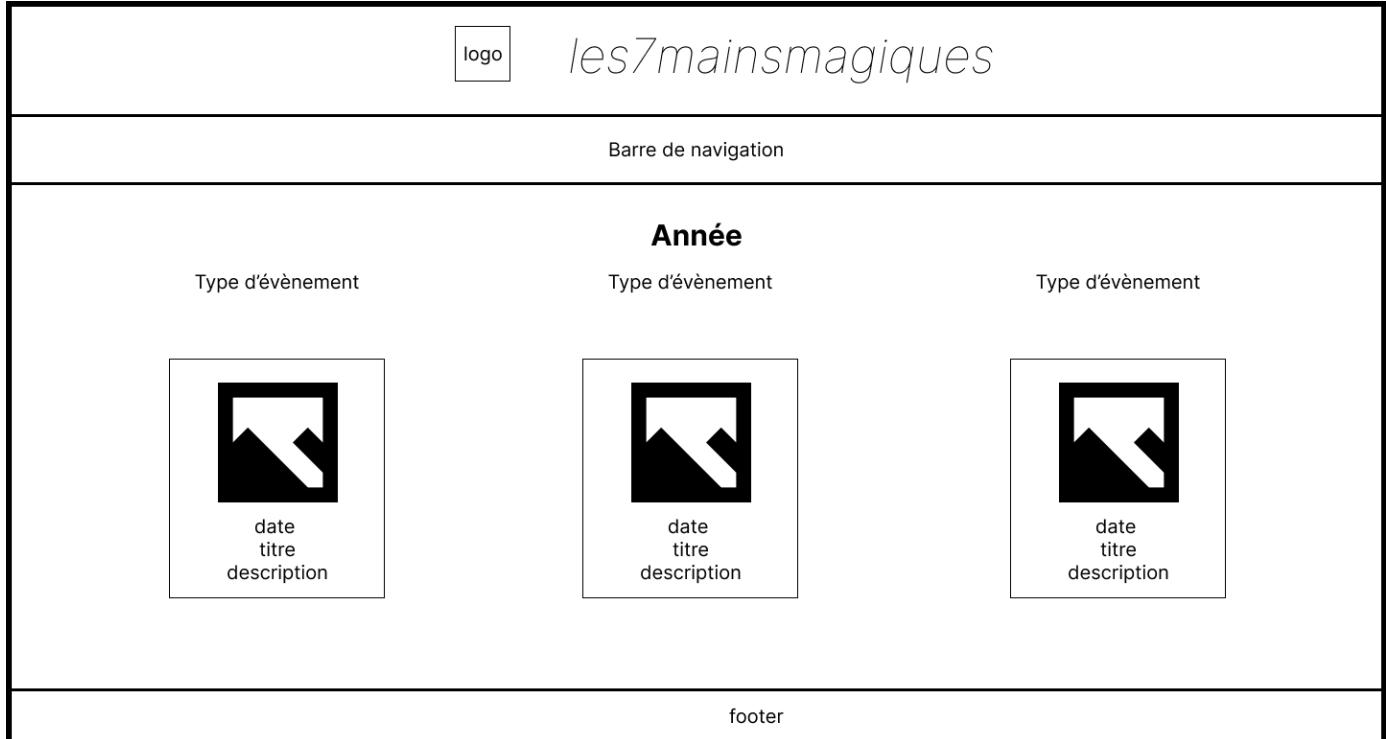
Suite à l'établissement du cahier des charges, j'ai initié mon projet en concevant une maquette. Cette étape m'a offert une visualisation concrète des différents éléments et des fonctions prévues pour l'application. L'objectif sous-jacent à ce processus de maquettage était de donner forme aux idées tout en établissant une cohérence visuelle et conceptuelle, créant ainsi un guide clair pour les étapes suivantes du développement.

L'utilisation de figma m'a permis d'élaborer une maquette interactive afin de naviguer rapidement sur le futur site .

## Accueil



## Affichage des événements



## Réalisation d'une interface utilisateur web statique et adaptable :

La création de mon interface web statique et adaptable a été concrétisée à travers la conception d'une interface utilisateur visuellement attrayante et fonctionnelle, adaptée à différentes tailles d'écrans à l'aide des langages HTML5 et CSS3 . Cette réalisation intègre des éléments graphiques soigneusement élaborés et garantit une expérience utilisateur optimale, quels que soient les dispositifs utilisés.

Version bureau de la page d'accueil :

The screenshot shows the homepage of the website [les7mainsmagiques](http://expo.test). At the top, there is a navigation bar with four tabs: "Présentation" (selected), "Programmation", "Galerie", and "Contact". Below the navigation bar is a large banner featuring a colorful abstract painting with black, white, red, and yellow elements. A white callout box is overlaid on the banner, containing the following text:

Crée en 2007, l'association *les7mainsmagiques*, loi de 1901, propose tout au long de l'année des visites dans les lieux culturels (galeries, musées, fondations, théâtres ...) et organise des expositions d'art contemporain. Elle a pour vocation de partager amitié et émotions autour d'activités présentant un intérêt culturel.

At the bottom of the page, there is a footer with the text "2021 Association des Nouveaux Artistes | Tous droits réservés." and a small link "expo/test/controllers/gallery\_controller.php".

Version android :



## Développer une interface utilisateur web dynamique :

J'ai utilisé le langage javascript afin de rendre mon interface dynamique :

- au passage de la souris les boutons change de couleur ;
- lorsque l'utilisateur descend sur la page, la barre de navigation apparaît à nouveau afin que l'utilisateur puisse changer de page ;
- la barre de navigation permet à l'utilisateur de savoir sur quelle page il se trouve .
- sur la page "Ajouter un événement", lorsque le type d'événement sélectionné est une exposition, les champs pour les dates deviennent visibles, ainsi qu'un emplacement est prévu pour télécharger l'image qui servira d'affiche. Lorsque les autres types d'événements sont choisis, seul le champ pour une date apparaît .

J'ai utilisé bootstrap pour l'apparition d'une modal : lorsqu'un utilisateur souhaite s'inscrire une fenêtre apparaît afin qu'il puisse enregistrer son email.

## Développer une interface utilisateur avec une solution de gestion de contenu :

En adoptant l'architecture Modèle-Vue-Contrôleur (MVC), la création d'une interface utilisateur s'est avérée cruciale pour la présentation des données à partir des modèles à travers les divers contrôleurs. Cette approche architecturale facilite également la mise en œuvre de la Programmation Orientée Objet (POO). Celle-ci favorise la modularité, la réutilisation du code, et une approche plus intuitive de la modélisation des problèmes complexes. Les concepts de la POO m'aident à organiser mon code de manière plus structurée et à concevoir un site plus flexible et évolutif.

La barre de navigation offre un accès aux diverses sections de l'application, et elle s'ajuste si l'administrateur est connecté ou non .

Utilisateur :

 les7mainsmagiques

Présentation      Programmation      Galerie      Contact

## Année 2023

**Exposition**



Du 27 Novembre au 30 Novembre

**Notre automne**  
Alfortville  
Les feuilles qui flétrissent, un spectacle qui nous ravit.

**Sortie**



03 Décembre

**Promenade aux jardins**  
Jardin Partagé  
Rendez-vous à 15h devant les jardins

[Participer](#)

**Assemblée générale**



30 Novembre

**Assemblée de décembre**  
Paris  
Rendez-vous au parc à 15h

[Participer](#)

Administrateur :

Présentation      Programmation      Galerie      Contact      

## Ajouter un événement

## Année 2023

**Exposition**

[Modifier](#)    [Supprimer](#)



Du 27 Novembre au 30 Novembre

**Notre automne**  
Alfortville  
Les feuilles qui flétrissent, un spectacle qui nous ravit.

[Ajouter / supprimer des photos](#)

**Sortie**

[Modifier](#)    [Supprimer](#)



03 Décembre

**Promenade aux jardins**  
Jardin Partagé  
Rendez-vous à 15h devant les jardins

[Ajouter / supprimer des photos](#)

Participants :

**Assemblée générale**

[Modifier](#)    [Supprimer](#)



30 Novembre

**Assemblée de décembre**  
Paris  
Rendez-vous au parc à 15h

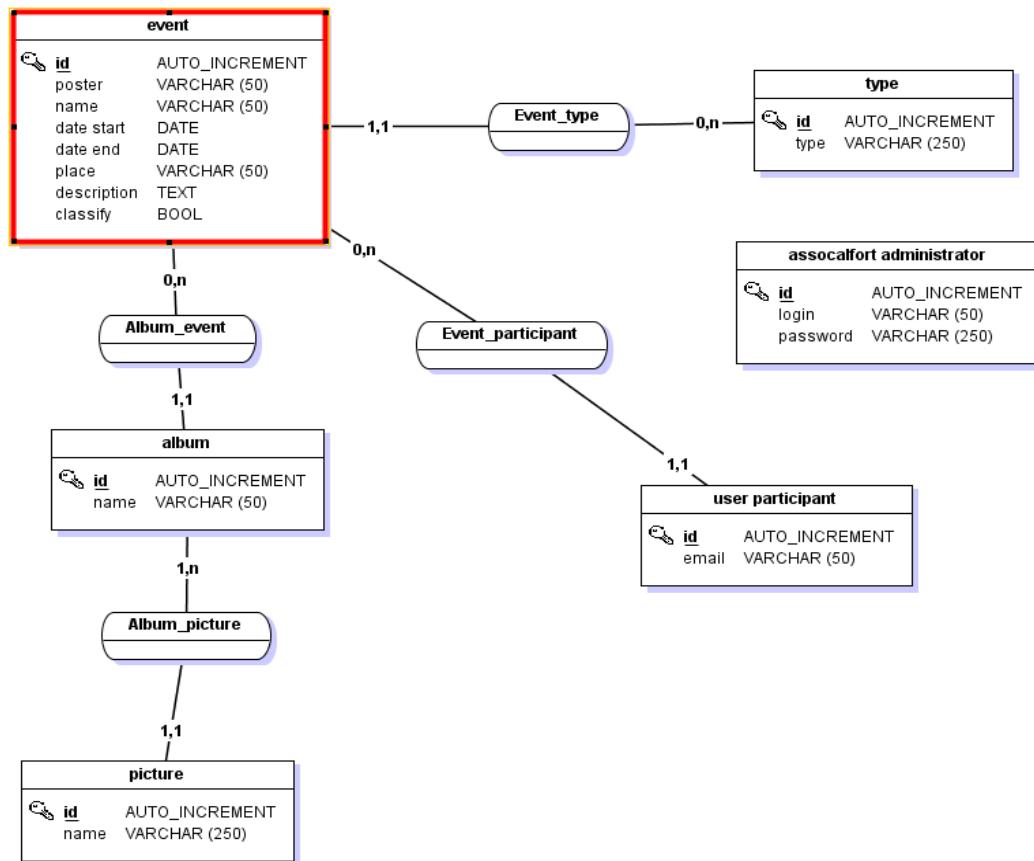
[Ajouter / supprimer des photos](#)

Participants :

## Création d'une base de données :

Après avoir défini les diverses fonctionnalités et achevé le processus de maquettage, il a été essentiel de mettre en place la base de données, celle-ci étant destinée à stocker l'ensemble des données de l'application. J'ai opté pour l'utilisation du logiciel JMerise afin de concevoir le modèle de mes entités, champs et cardinalités. Initialement, j'ai obtenu un Modèle Conceptuel de Données (MCD) que j'ai adapté au fil de l'évolution de mon projet en fonction de mes besoins spécifiques.

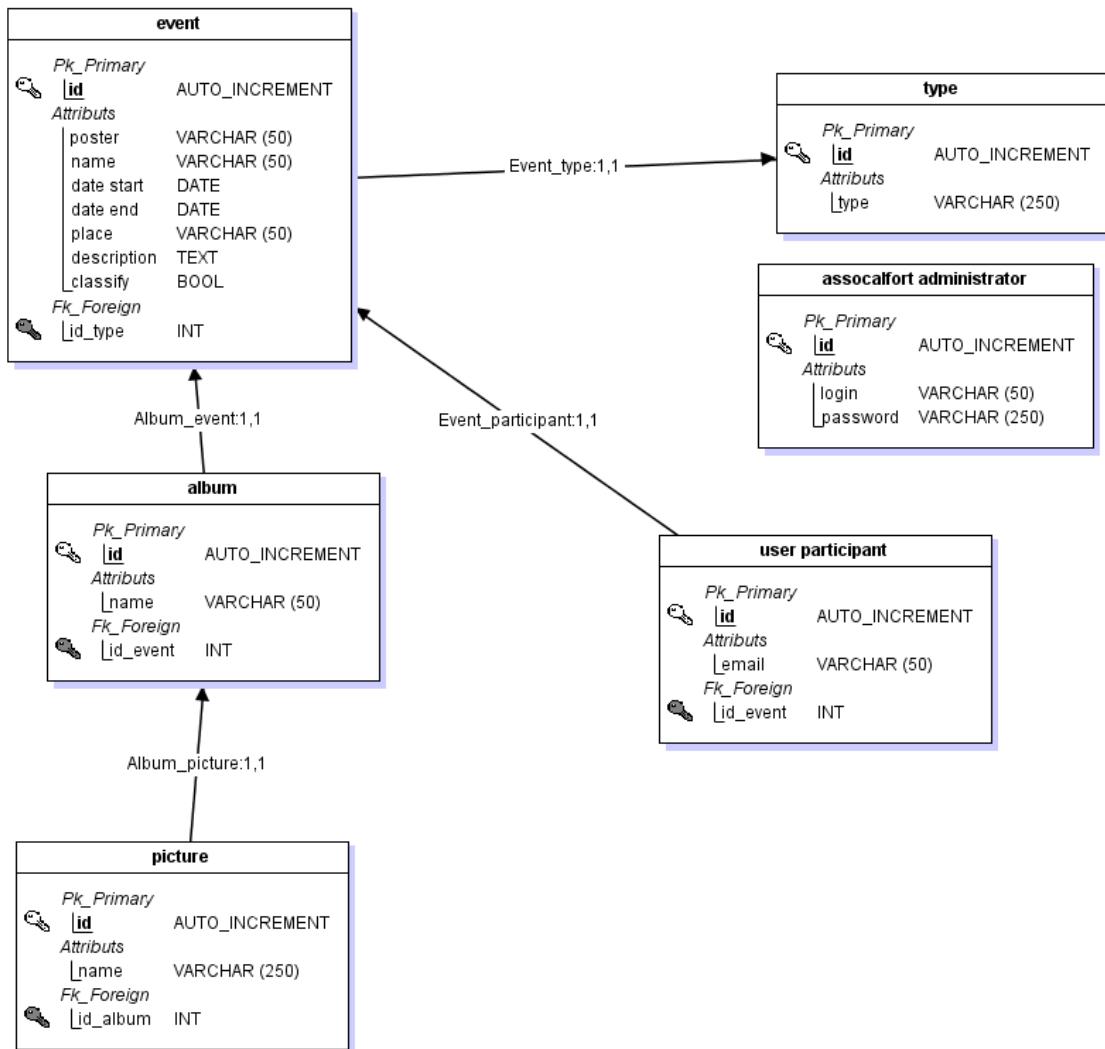
MCD :



Ensuite, après avoir élaboré mon Modèle Conceptuel de Données (MCD), j'ai tiré parti de la fonctionnalité de JMerise pour générer automatiquement mon Modèle Logique de Données (MLD). Cet outil s'appuie sur

les clés primaires et les cardinalités que j'ai définies, facilitant ainsi la création des clés étrangères dans les différentes tables.

MLD :



Après avoir élaboré mes Modèles Conceptuel de Données (MCD) et Logique de Données (MLD), j'ai utilisé une fonctionnalité de JMerise pour générer automatiquement le script SQL correspondant à ma conception.

```

#-----#
#      Script MySQL.
#-----#


#-----#
# Table: type
#-----#


CREATE TABLE type(
    id      Int Auto_increment NOT NULL ,
    type    Varchar (250) NOT NULL
           ,CONSTRAINT type_PK PRIMARY KEY (id)
)ENGINE=InnoDB;

#-----#
# Table: event
#-----#


CREATE TABLE event(
    id          Int Auto_increment NOT NULL ,
    poster      Varchar (50) NOT NULL ,
    name        Varchar (50) NOT NULL ,
    date_start  Date ,
    date_end    Date ,
    place       Varchar (50) ,
    description Text ,
    classify    Bool NOT NULL ,
    id_type     Int NOT NULL
           ,CONSTRAINT event_PK PRIMARY KEY (id)
           ,CONSTRAINT event_type_FK FOREIGN KEY (id_type) REFERENCES type(id)
)ENGINE=InnoDB;

#-----#
# Table: assocalfort administrator
#-----#


CREATE TABLE assocalfort_administrator(
    id      Int Auto_increment NOT NULL ,
    login   Varchar (50) NOT NULL ,
    password Varchar (250) NOT NULL
           ,CONSTRAINT assocalfort_administrator_PK PRIMARY KEY (id)
)ENGINE=InnoDB;

#-----#
# Table: user participant
#-----#


CREATE TABLE user_participant(
    id      Int Auto_increment NOT NULL ,
    email   Varchar (50) NOT NULL ,
    id_event Int NOT NULL
           ,CONSTRAINT user_participant_PK PRIMARY KEY (id)
           ,CONSTRAINT user_participant_event_FK FOREIGN KEY (id_event) REFERENCES event(id)
)ENGINE=InnoDB;

```

```

#-----#
# Table: album
#-----#
CREATE TABLE album(
    id      Int  Auto_increment NOT NULL ,
    name    Varchar (50) NOT NULL ,
    id_event Int NOT NULL
    ,CONSTRAINT album_PK PRIMARY KEY (id)

    ,CONSTRAINT album_event_FK FOREIGN KEY (id_event) REFERENCES event(id)
)ENGINE=InnoDB;

#-----#
# Table: picture
#-----#
CREATE TABLE picture(
    id      Int  Auto_increment NOT NULL ,
    name    Varchar (250) NOT NULL ,
    id_album Int NOT NULL
    ,CONSTRAINT picture_PK PRIMARY KEY (id)

    ,CONSTRAINT picture_album_FK FOREIGN KEY (id_album) REFERENCES album(id)
)ENGINE=InnoDB;

```

Evidemment le mot de passe entrer manuellement a été hasher

	id	login	password
▶	1	donati4@orange.fr	\$2y\$10\$MawioXLoFL35y0xkq5EY/Y...
*	NULL	NULL	NULL

## Développer des composants d'accès aux données :

Dans une architecture MVC, chaque modèle correspond à une table de la base de données et doit établir une connexion à chaque exécution d'une méthode pour accéder aux données. Ainsi, j'ai développé une classe dédiée regroupant tous les éléments essentiels permettant la connexion à la base de données.

```
// Je définie les constantes de connexion à la base de données
define ("HOST" , "localhost");
define ("USER" , "root");
define ("PASS" , "8s3uY.(sSmJa8s4p");
define ("DBNAME" , "les7mainsmagiques");
```

Je crée une classe « database» qui sera commune à toutes les autres classes requérant une connexion à la base de données.

```
// Je créer une classe database qui me permettra de me connecter à la base de données
class database {
    public static function getDatabase(){
        $dbh = 'mysql:host=' . HOST .';dbname=' . DBNAME .';charset=utf8';
        // Je créer un objet PDO qui me permettra de me connecter à la base de données en utilisant
        // les constantes définies dans config.php ainsi que la méthode try catch pour gérer les erreurs
        try {
            $db = new PDO($dbh, USER, PASS);
            $db->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
            return $db;
        } catch (PDOException $e) {
            echo 'Connexion failed : ' . $e->getMessage();
        }
    }
}
```

## Développer la partie back-end d'une application web ou web mobile :

Comme mentionné précédemment, dans une architecture MVC, un modèle correspond à une table de la base de données. Dans notre application, les modèles font usage de la classe "Database" pour établir une connexion et proposent diverses méthodes pour créer, lire, mettre à jour et supprimer les données respectives. Afin de les utiliser, les modèles sont directement instanciés dans les contrôleurs des différentes vues. En effet, les vues ont pour rôle d'afficher des données et d'interagir avec l'utilisateur. À cette fin, chaque vue est associée à un contrôleur respectif.

Je vérifie que l'administrateur est connecté via un « !isset(\$\_SESSION['admin']) » sinon je redirige l'utilisateur vers la page d'accueil .

Les modèles et le contrôleur sont inclus via un « require\_once » pour assurer la disponibilité de ces composants essentiels dans le cadre de l'architecture MVC.

```
// Je vérifie que l'utilisateur est bien connecté en tant qu'admin
// sinon je le redirige vers la page d'accueil
session_start();

if (!isset($_SESSION['admin'])) {
    header('Location: ../index.php');
    exit;
}

// J'inclus les fichiers nécessaires à la page
// afin de pouvoir utiliser leurs classes, fonctions, etc...
require_once '../config.php';
require_once '../helpers/database.php';
require_once '../helpers/form.php';

require_once '../models/events.php';
require_once '../models/type.php';
require_once '../models/album.php';
require_once '../models/picture.php';
```

La vue est intégré à l'aide d'un « include » à la fin du contrôleur afin que la logique soit utilisable dans la vue

```
129
130
131     include '../views/add_event.php';
132
```

Je crée une nouvelle instance à l'aide du modèle correspondant

```
<?php

// Je crée une classe Event qui possède des attributs et des fonctions
// Cette classe va me permettre de gérer les événements
// Je vais pouvoir ajouter, modifier, supprimer, récupérer des événements
// dans ma base de données
class Event {

    // Je crée les attributs de la classe Event
    private int $id;
    private string $poster;
    private string $name;
    private int $type;
    private string $dateStart;
    private string $dateEnd;
    private string $place;
    private string $description;
    private string $album;
    private string $picture;
```

Ici , je prépare ma requête à l'aide de "bindValue" et grâce à ma fonction "secureData" je me protège des injection SQL

```
// Je crée une fonction getEvents() qui va me permettre de récupérer tous
// les événements en fonction de l'id et qui retourne un tableau associatif
public static function getEventById(int $id): array
{
    // J'utilise la méthode try catch pour essayer d'exécuter mon code
    try {
        // Je me connecte à ma base de données
        $db = database::getDatabase();
        // Je crée ma requête SQL avec un alias pour les champs nécessaires
        $sql = "SELECT
            `event`.`id` `event_id`,
            `poster`,
            `event`.`name` `event_name`,
            `date_start`,
            `date_end`,
            `place`,
            `description`,
            `id_type` `type_id`
        FROM
            `event`
        WHERE
            `event`.`id` = :id";
        // Je prépare ma requête SQL
        $query = $db->prepare($sql);
        // J'associe les valeurs reçues en paramètre aux champs de la table
        $query->bindValue(':id', form::secureData($id), PDO::PARAM_INT);
        // J'exécute ma requête SQL
        $query->execute();
        // Je retourne le résultat de ma requête SQL sous forme de tableau associatif
        return $query->fetch(PDO::FETCH_ASSOC);
    }
```

La fonction "secureData" qui m'aide à me protéger des injections SQL

```
// Création d'une fonction pour sécuriser les données envoyées par l'utilisateur
public static function secureData($data)
{
    // Supprime les espaces inutiles, les antislashes et les caractères spéciaux
    $data = trim($data);
    $data = stripslashes($data);
    $data = htmlspecialchars($data);
    // Retourne les données sécurisées
    return $data;
}
```

Puis j'appelle la méthode pour l'utiliser

```
$query->bindValue(':poster', form::secureData($poster) , PDO::PARAM_STR);
$query->bindValue(':name', form::secureData($name) , PDO::PARAM_STR);
$query->bindValue(':type', form::secureData($type) , PDO::PARAM_INT);
$query->bindValue(':dateStart', form::secureData($dateStart) , PDO::PARAM_STR);
$query->bindValue(':dateEnd', form::secureData($dateEnd) , PDO::PARAM_STR);
$query->bindValue(':place', form::secureData($place) , PDO::PARAM_STR);
$query->bindValue(':description', form::secureData($description) , PDO::PARAM_STR);
```

# III. Réalisation

## Création d'un événement :

Mise en place d'une vue comprenant un formulaire destiné à recueillir les informations nécessaire pour créer un événement .

### Ajouter un évènement

Champs obligatoire\*

Nom de l'évènement : \*

Ex. Exposition d'hiver

Type d'évènement : \*

Selectionner ▾

Lieu de l'évènement : \*

Ex. Alfortville

Description de l'évènement : (facultatif)

Ex. Champignons en carton

Ajouter l'évènement

Retour à la page galerie

Retour à la page programmation

Création d'un contrôleur afin de préparer l'enregistrement des inputs en réalisant des contrôles grâce à des conditions ou des regex . Évidemment je vérifie s'il existe une session administrateur afin de sécuriser l'accès

```
// Je vérifie que l'utilisateur est bien connecté en tant qu'admin
// sinon je le redirige vers la page d'accueil
session_start();

if (!isset($_SESSION['admin'])) {
    header('Location: ../index.php');
    exit;
}

// Je créer un tableau $error qui contiendra les messages d'erreur
$error = [];

// Je créer une variable $showform qui me permettra d'afficher ou non le formulaire
$showform = true;

// Je vérifie que le formulaire a bien été soumis
if ($_SERVER['REQUEST_METHOD'] == 'POST') {

    // Je créer une variable $album_name qui contient le nom de l'album en minuscule et sans accent
    $album_name = strtolower(Form::noAccent($_POST['event_name']));

    // Je vérifie que les champs obligatoires sont remplis
    // Si un champ n'est pas remplis, j'ajoute un message d'erreur dans le tableau $error
    // Sinon je récupère la valeur du champ dans une variable

    // Je vérifie que le nom de l'évènement est bien renseigné
    if (empty($_POST['event_name'])) {
        $error['event_name'] = 'Le nom de l\'évènement est obligatoire';
    }
    // Je vérifie que le nom de l'évènement n'existe pas déjà dans la base de données
    else if (Event::getNameEvent($_POST['event_name'])) [
        $error['event_name'] = 'Ce nom d\'évènement existe déjà';
    ]
    // Je vérifie que le nom de l'évènement n'existe pas déjà dans la base de données des albums
    else if (Album::checkAlbumName($album_name)) {
        $error['event_name'] = 'Ce nom d\'évènement existe déjà';
    } else {
        $name = $_POST['event_name'];
    }
}
```

Dans le modèle de ma table "events" , je crée une méthode pour ajouter l'événement à la base de données, que j'utiliserai une fois les informations validées

```

// Je crée une fonction addEvent() qui va me permettre d'ajouter un événement et qui retourne un booléen
public static function addEvent (string $poster, string $name, int $type, string $dateStart, string $dateEnd, string $place, string $description): bool
{
    // J'utilise la méthode try catch pour essayer d'exécuter mon code
    // Si une erreur se produit, elle sera récupérée par le catch
    try {
        // Je me connecte à ma base de données
        $db = database::getDatabase();
        // Je crée ma requête SQL
        $sql = "INSERT INTO `event`
        (`poster`, `name`, `id_type`, `date_start`,
        `date_end`, `place`, `description`)
        VALUES
        (:poster, :name, :type, :dateStart,
        :dateEnd, :place, :description)";

        // Je prépare ma requête SQL
        $query = $db->prepare($sql);
        // J'associe les valeurs reçues en paramètre aux champs de la table
        // en utilisant la méthode bindValue()
        // Je sécurise les données reçues en paramètre avec la méthode secureData()
        // Je précise le type de données reçues en paramètre avec la méthode PDO::PARAM_
        $query->bindValue(':poster', form::secureData($poster) , PDO::PARAM_STR);
        $query->bindValue(':name', form::secureData($name) , PDO::PARAM_STR);
        $query->bindValue(':type', form::secureData($type) , PDO::PARAM_INT);
        $query->bindValue(':dateStart', form::secureData($dateStart) , PDO::PARAM_STR);
        $query->bindValue(':dateEnd', form::secureData($dateEnd) , PDO::PARAM_STR);
        $query->bindValue(':place', form::secureData($place) , PDO::PARAM_STR);
        $query->bindValue(':description', form::secureData($description) , PDO::PARAM_STR);

        // J'exécute ma requête SQL
        return $query->execute();
    }
    // Si une erreur se produit, elle sera récupérée par le catch
    catch (PDOException $e) {
        echo 'Erreur : ' . $e->getMessage();
        return false;
    }
}

```

Ensuite j'offre la possibilité à l'administrateur d'ajouter des photos , lorsque la variable \$showform prend la valeur false



[Ajouter les photos de l'évènement](#) [Retour à l'accueil](#)

Dans le contrôleur je vérifie qu'il existe une session administrateur puis si le fichier télécharger est bien une image et qu'elle ne dépasse pas 1 mo

```
// Je démarre la session
session_start();
// Si la session administrateur n'existe pas je redirige vers la page d'accueil
if (!isset($_SESSION['admin'])) {
    header('Location: ../index.php');
    exit;
}
```

```

// Je créer un nom d'album en fonction du nom de l'évènement
$album_name = strtolower(Form::noAccent($event['event_name']));
// Je vérifie que la méthode POST existe et que le formulaire a été soumis
if ($_SERVER['REQUEST_METHOD'] === 'POST') {

    // Je vérifie que le champ picture existe et n'est pas vide
    if (isset($_POST["id_picture"])) {
        // Je supprime la photo de l'album
        Picture::deletePicture($_POST["id_picture"]);
    }
    // Je vérifie que le l'input files existe et n'est pas vide
    else if (!empty($_FILES) && $_FILES["picture"]["error"] != 4) {
        // J'utilise la classe finfo pour récupérer le type MIME du fichier
        $tfile = new finfo(FILEINFO_MIME);
        // Je vérifie que le fichier est bien une image
        if (!str_contains($tfile->file($_FILES['picture']['tmp_name']), "image")) {
            $error['picture'] = "Le format doit être de type image (jpg, jpeg, png)";
        } else if (!str_contains($_FILES["picture"]["type"], "image")) {
            $error['picture'] = "Le format doit être de type image (jpg, jpeg, png)";
        }
        // Je vérifie que le fichier ne dépasse pas 1Mo
        else if ($_FILES["picture"]["size"] > 1048576) {
            $error['picture'] = "Le justificatif ne doit pas dépasser 1Mo";
        }
        // Je vérifie qu'il n'y a pas d'erreur
        if (empty($error['picture'])) {
            // Si le fichier possède une extension jpg, jpeg ou png je supprime le nom de l'extension
            // pour ajouter le nom de l'album et remettre le nom de l'extension à la fin afin d'obtenir un nom unique
            // de type : nom_photo.nom_album.jpg
            if (str_contains($_FILES["picture"]["name"], '.jpg')) {
                $picture = $_FILES["picture"]["name"] . '.' . strtolower(Form::noAccent($event["event_name"]));
                $picture = str_replace('.jpg', '', $picture);
                $picture = $_FILES['picture'][name] . '.' . $album_name;
                $picture = $picture . ".jpg";
            } else if (str_contains($_FILES["picture"]["name"], '.png')) {
                $picture = $_FILES["picture"]["name"] . '.' . strtolower(Form::noAccent($event["event_name"]));
                $picture = str_replace('.png', '', $picture);
                $picture = $_FILES['picture'][name] . '.' . $album_name;
                $picture = $picture . ".png";
            } else if (str_contains($_FILES["picture"]["name"], '.jpeg')) {
                $picture = $_FILES["picture"]["name"] . '.' . strtolower(Form::noAccent($event["event_name"]));
                $picture = str_replace('.jpeg', '', $picture);
                $picture = $_FILES['picture'][name] . '.' . $album_name;
                $picture = $picture . ".jpeg";
            } else {
                $error['event_picture'] = "Le format doit être de type image (jpg, jpeg, png)";
            }
        }
    }
}

```

## Afficher un événement :

Dans la vue correspondante , je crée des cards pour pouvoir afficher tous les événements à venir ou passée

 les7mainsmagiques

Présentation      Programmation      Galerie      Contact

Année 2023

**Exposition**  
  
Du 27 Novembre au 30 Novembre

**Notre automne**  
Alfortville  
Les feuilles qui flétrissent, un spectacle qui nous ravit.

**Sortie**  
  
03 Décembre

**Promenade aux jardins**  
Jardin Partagé  
Rendez-vous à 15h devant les jardins  
[Participer](#)

**Assemblée générale**  
*Assemblée Générale*  
  
30 Novembre

**Assemblée de décembre**  
Paris  
Rendez-vous au parc à 15h  
[Participer](#)

Présentation      Programmation      Galerie      Contact

2023

  
**Fête des morts**  
Du 01 Novembre au 07 Novembre  
À Le Havre

*Assemblée Générale*  
  
**Assemblée de novembre**  
22 Novembre  
À Paris

  
**Sortie novembre**  
23 Novembre  
À Alfortville

2022



Dans le modèle, je crée deux méthodes : une pour récupérer les événements antérieurs à la date du jour et une seconde pour ceux ultérieurs qui seront respectivement placés dans la page galerie et la page programmation

```
// Je crée une fonction getEvents() qui va me permettre de récupérer tous les événements
public static function getNewEvents(string $year, int $type): array
{
    // J'utilise la méthode try catch pour essayer d'exécuter mon code
    try {
        // Je me connecte à ma base de données
        $db = database::getDatabase();
        // Je modifie la valeur de $year pour qu'elle corresponde à la requête SQL
        $year = $year . '%';
        // Je crée ma requête SQL avec un alias pour les champs nécessaires
        $sql = "SELECT
            `event`.`id` `event_id`,
            `poster`,
            `event`.`name` `event_name`,
            `date_start`,
            `date_end`,
            `place`,
            `description`,
            `type`.`id` `type_id`,
            `type`.`type` `type_name`,
            `album`.`id` `album_id`,
            `album`.`name` `album_name`
        FROM
            `event`
            INNER JOIN
            `type` ON `id_type` = `type`.`id`
            LEFT JOIN
            `album` ON `id_event` = `event`.`id`
        WHERE
            // Je prépare ma requête SQL
            $query = $db->prepare($sql);
            // J'associe les valeurs reçues en paramètre aux champs de la table
            $query->bindValue(':year', form::secureData($year), PDO::PARAM_STR);
            // J'exécute ma requête SQL
            $query->execute();
            // Je retourne le résultat de ma requête SQL sous forme de tableau associatif
            return $query->fetchAll(PDO::FETCH_ASSOC);
    }
    // Si une erreur se produit, elle sera récupérée par le catch
    catch (PDOException $e) {
        echo 'Erreur : ' . $e->getMessage();
        return [];
    }
}
```

## Modifier un événement :

Dans la vue correspondante, je conçois un formulaire pour présenter les données préalablement enregistrées, autorisant ainsi l'utilisateur à les modifier, facilitant ainsi le processus de mise à jour.

# Modifier un évènement

Champs obligatoire \*

Nom de l'évènement : \*

Fête des morts

Type d'évènement : \*

Exposition

Lieu de l'évènement : \*

Le Havre

Dates de l'évènement : \*

Du :

01/11/2023



Au :

07/11/2023



Description de l'évènement : (facultatif)

Veillons sur nos morts comme ils  
veillent sur nous .

Affiche actuel :



Photo d'affiche \*

Choisir un fichier Aucun fichier choisi

Je crée un contrôleur qui va confirmer la connexion de l'administrateur ainsi que de vérifier les informations modifiés pour éviter toute erreur , si une entrée est identique à une existante dans la base de donnée pour éviter les doublons

```
<?php
// Vérification de l'existence d'une session
session_start();
// Si la session n'existe pas on redirige vers la page d'accueil
if (!isset($_SESSION['admin'])) {
    header('Location: ../index.php');
    exit;
}

// Inclusion des fichiers de configuration et des différents modèles
require_once '../config.php';
require_once '../helpers/database.php';
require_once '../helpers/form.php';

require_once '../models/events.php';
require_once '../models/type.php';
require_once '../models/album.php';
require_once '../models/picture.php';

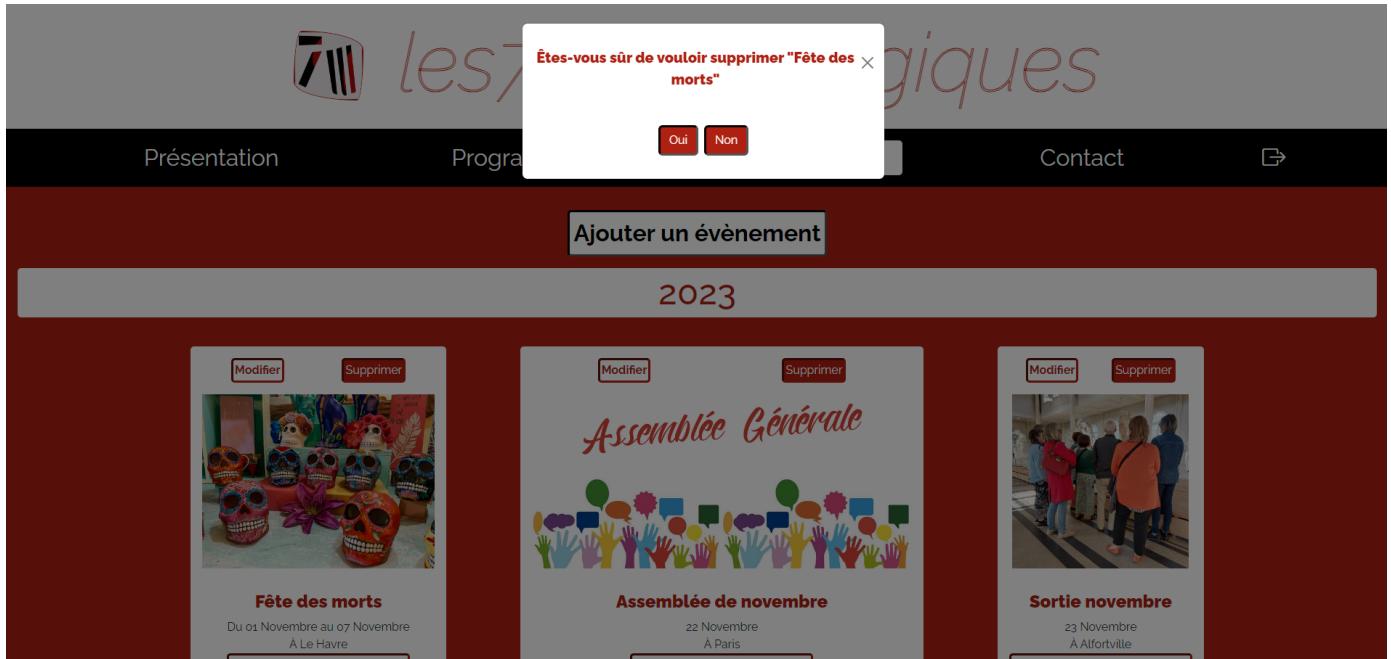
// Vérification de l'id de l'évènement
if (isset($_GET["id"]) && !empty($_GET["id"])) {
    // Vérification que l'id est bien un nombre sinon redirection vers la page d'accueil
    if (!ctype_digit($_GET["id"])) {
        header('Location: ../index.php');
        exit;
    } else {
        // Création d'une variable pour stocker l'id de l'évènement
        $id = strip_tags($_GET["id"]);
        // Appel de la méthode getEventById pour récupérer l'évènement
        $event = Event::getEventById($id);
        // Vérification que l'évènement existe sinon redirection vers la page d'accueil
        if ($event === false) {
            header('Location: ../index.php');
            exit;
        }
    }
}
// Si l'id n'est pas renseigné dans l'url redirection vers la page d'accueil
else {
    header('Location: ../index.php');
    exit;
}
```

Dans le modèle correspondant, je crée une méthode pour mettre à jour selon l'id de l'enregistrement qualité.

```
// Je crée une fonction updateEvent() qui va me permettre de modifier un événement et qui retourne un booléen
public static function updateEvent(int $id, string $poster, string $name, int $type, string $dateStart, string $dateEnd, string $place, string $description): bool
{
    // J'utilise la méthode try catch pour essayer d'exécuter mon code
    try {
        // Je me connecte à ma base de données
        $db = database::getDatabase();
        // Je crée ma requête SQL
        $sql = "UPDATE `event`"
            . "SET"
            . "    `poster` = :poster,"
            . "    `name` = :name,"
            . "    `id_type` = :type,"
            . "    `date_start` = :dateStart,"
            . "    `date_end` = :dateEnd,"
            . "    `place` = :place,"
            . "    `description` = :description"
            . "WHERE"
            . "    `id` = :id";
        // Je prépare ma requête SQL
        $query = $db->prepare($sql);
        // J'associe les valeurs reçues en paramètre aux champs de la table
        $query->bindValue(':id', form::secureData($id) , PDO::PARAM_INT);
        $query->bindValue(':poster', form::secureData($poster) , PDO::PARAM_STR);
        $query->bindValue(':name', form::secureData($name) , PDO::PARAM_STR);
        $query->bindValue(':type', form::secureData($type) , PDO::PARAM_INT);
        $query->bindValue(':dateStart', form::secureData($dateStart) , PDO::PARAM_STR);
        $query->bindValue(':dateEnd', form::secureData($dateEnd) , PDO::PARAM_STR);
        $query->bindValue(':place', form::secureData($place) , PDO::PARAM_STR);
        $query->bindValue(':description', form::secureData($description) , PDO::PARAM_STR);
        // J'exécute ma requête SQL
        return $query->execute();
    }
    // Si une erreur se produit, elle sera récupérée par le catch
    catch (PDOException $e) {
        echo 'Erreur : ' . $e->getMessage();
        return false;
    }
}
```

## Supprimer un événement :

Afin de finaliser la suppression d'un événement, j'ai intégré une modale dans la vue pour confirmer et valider l'opération de suppression.



```

<button type="button" class="cancel" data-bs-toggle="modal" data-bs-target="#modal<?= $event["event_id"] ?>">
    Supprimer
</button>

<!-- Modal -->
<div class="modal fade" id="modal<?= $event["event_id"] ?>" tabindex="-1" aria-labelledby="exampleModalLabel" aria-hidden="true">
    <div class="modal-dialog">
        <div class="modal-content">
            <div class="modal-header delete_confirm">
                <h4 class="modal-title fs-5" id="exampleModalLabel">Êtes-vous sûr de vouloir supprimer "<?= $event["event_name"] ?>"</h4>
                <button type="button" class="btn-close" data-bs-dismiss="modal" aria-label="Close"></button>
            </div>
            <div class="modal-footer button_confirm">
                <form method="post">
                    <input type="hidden" name="event_id" value="<?= $event["event_id"] ?>">
                    <button name="delete" class="btn btn-primary delete_new_event">Oui</button>
                </form>
                <button type="button" class="btn btn-secondary delete_new_event" data-bs-dismiss="modal">Non</button>
            </div>
        </div>
    </div>
</div>

```

Dans le contrôleur, je récupère le choix de l'administrateur pour supprimer l'événement

```

// Je vérifie si le delete de la méthode POST est demandé
if (isset($_POST["delete"])) {
    // Je vérifie si l'album existe
    if (Album::existAlbum($_POST["event_id"])) {
        // Je récupère le nom de l'album
        $folder = Album::getAlbumById($_POST["event_id"])[0]["name"];
        // Je vérifie si le dossier existe
        if (is_dir($folder)) {
            // Je récupère les images du dossier
            $images = glob('../assets/img/' . $folder . '/*');
            // Je supprime les images du dossier
            foreach ($images as $image) {
                if (is_file($image)) {
                    unlink($image);
                }
            }
            // S'il n'y a plus d'images dans le dossier, je supprime le dossier
            if (empty($images)) {
                rmdir('../assets/img/' . $folder);
            }
        }
        // Je supprime l'événement
        Event::deleteEvent($_POST["event_id"]);
    }
    // Si le dossier n'existe pas, je supprime l'événement
    else {
        Event::deleteEvent($_POST["event_id"]);
    }
}
// Si l'album n'existe pas, je supprime l'événement
else {
    Event::deleteEvent($_POST["event_id"]);
}

```

Je crée une méthode afin de supprimer l'événement de la base de données selon les informations envoyé au contrôleur

```

// Je crée une fonction deleteEvent() qui va me permettre de supprimer
// un événement en fonction de son id et qui retourne un booléen
public static function deleteEvent(int $id): bool
{
    // J'utilise la méthode try catch pour essayer d'exécuter mon code
    try {
        // Je me connecte à ma base de données
        $db = database::getDatabase();
        // Je crée ma requête SQL
        $sql = "DELETE FROM `event` WHERE `id` = :id";
        // Je prépare ma requête SQL
        $query = $db->prepare($sql);
        // J'associe les valeurs reçues en paramètre aux champs de la table
        $query->bindValue(':id', form::secureData($id), PDO::PARAM_INT);
        // J'exécute ma requête SQL
        return $query->execute();
    }
    // Si une erreur se produit, elle sera récupérée par le catch
    catch (PDOException $e) {
        echo 'Erreur : ' . $e->getMessage();
        return false;
    }
}

```

## Anglais de rigueur : Une communauté importante orientée vers le partage

La situation de travail nécessitant une recherche en anglais pourrait être la suivante : en tant que développeur web, il peut être nécessaire de consulter des ressources, tutoriels, documentations ou forums en anglais pour obtenir des informations détaillées sur les dernières technologies, les meilleures pratiques de développement, ou pour résoudre des problèmes spécifiques liés au code. La recherche en anglais permet d'accéder à un large éventail de connaissances dans le domaine du développement web, qui est souvent abondamment partagé dans cette langue sur Internet.

Lors de mes recherches j'ai visité ce site afin d'apprendre à envoyer un email aux utilisateurs qui souhaite s'inscrire aux événements

# 1. Sending an email using the PHP Web Server

PHP offers to send emails directly from the hosted web server by providing an inbuilt function mail(). There are pros and cons to using the inbuilt mail() function. Let us look at this in detail.

## Pros

1. It is very straightforward to use as it requires calling one function with the required parameters.
2. It is in-built with PHP, so it provides tight integration to the PHP ecosystem that ensures plug-and-play compatibility across PHP Frameworks.
3. It helps send internal text-based emails easily.

## Cons

1. It has no support for email attachments.
2. It does not support external SMTP Authentication or DKIM signatures. It may cause your emails to be flagged as spam by your recipient email clients.
3. It is not suitable for many users' event-driven alerts as it opens and closes one SMTP Socket Connection per email. Therefore, bulk emails are inefficient.

## Demonstration

However, if you feel like mail() suits your needs, look at the code shown below to send an email using mail().

The function shown above is the PHP mail function. As you can see, it accepts five parameters. But you can send an email using the four parameters discussed below.

1. \$to\_email\_address: The recipient of the email. This address must be compliant with the [RFC-2822 Standards](#).
2. \$subject: The subject of the email.
3. \$message: The email body (Text/ HTML). Each line in the email must not exceed 70 characters, and a line must be separated using a new line token (\n).
4. \$headers: The headers of the email. The "from" header is mandatory, and the "Content-Type" header is only compulsory when sending a templated email.

Et voilà ma traduction :

PHP propose d'envoyer des emails directement depuis le serveur web hôte en fournissant la fonction intégrer *mail()*.

Il y a des avantages et des inconvénients pour utiliser la fonction intégrer *mail()* . Voyons cela dans le détail :

### **Avantages**

1. Son utilisation est très simple, nécessitant simplement l'appel d'une fonction avec les paramètres requis.
2. C'est une fonction intégrée à PHP, elle fournit une intégration parfaite à l'écosystème php ce qui assure un prêt à l'emploi compatible sur tous les Framework PHP.
3. Elle aide à envoyer facilement des emails textuels de façon interne

### **Inconvénients**

1. Il n'y a pas de support pour joindre des fichiers à l'email .
2. Elle ne supporte pas les authentifications externes ou les signatures numériques . L'email peut être placé dans les spams par le client du destinataire de l'email .
3. Il n'est pas adapté aux alertes événementielles de nombreux utilisateurs, car il ouvre et ferme une connexion SMTP Socket par courrier électronique. Par conséquent, les courriels en masse sont inefficaces.

Cependant , si vous pensez que *mail()* est ce dont vous avez besoin , regardez le code ci-dessous pour envoyer un email utilisant *mail()* .

La fonction montrée ci-dessus est la fonction mail de PHP .Comme vous pouvez le constater , elle accepte cinq paramètres. Mais vous pouvez envoyer un email en utilisant les 4 paramètres ci-dessous :

1. \$to\_email\_address : le destinataire de l'e-mail . Cette adresse doit être conforme au standard RFC-2822
2. \$subject : l'objet de l'e-mail
3. \$message : le corps de l'e-mail . Chaque ligne de l'e-mail de doit pas dépasser 70 caractères, et chaque ligne doit être séparer en utilisant la marque (\n)
4. \$headers : l'entête de l'e-mail . Le "from" de l'entête est obligatoire , et le "type de contenu" de l'entête est seulement obligatoire lors de l'envoie d'un email modèle .

## Conclusion

Le site les7mainsmagiques est encore à ses débuts, mais les fonctionnalités d'ajout, d'affichage, de modification et de suppression d'événements sont déjà opérationnelles. Suite à un échange avec la présidente, de nouvelles fonctionnalités sont en cours de préparation, telles que la gestion d'un tableau des adhérents, la possibilité de les inscrire à des événements, voire même l'ajout d'une fonction de connexion pour les adhérents. Bien que des ajustements soient nécessaires en termes de style et d'amélioration de la modal d'inscription aux événements, le site est déjà fonctionnel.

Dans les futures versions, je prévois d'intégrer des fonctionnalités telles que l agrandissement d'images, la modification des détails d'un événement, l'archivage des événements, l'indication de l'accessibilité pour les personnes handicapées, et la possibilité d'ajouter de nouveaux types d'événements. Évidemment je compte aussi améliorer la sécurité , notamment lors du téléchargement de fichiers . Ces ajouts seront intégrés dans une mise à jour de l'outil, combinant la puissance de calcul d'un tableur avec l'ergonomie conviviale d'une application web.

Ce projet m'a permis de réaliser que le développement d'une application demande du temps et un investissement considérable, depuis la conception jusqu'à la réalisation et les sessions de débogage. Je vous remercie de m'avoir lu et écouté, et j'espère avoir suscité votre intérêt pour en savoir davantage sur l'association les7mainsmagiques.

Merci .