Panel de gestion des visiteurs

Projet AP Semestre 2 - 2024/2025

Gestion des données du musé

La base de données du musée gère les entrées et sorties des visiteurs, limitées à 50 personnes simultanées, et enregistre les détails des expositions permanentes et temporaires.

Informations du document:

Rédacteurs : Evan MANSET, Gaspard TOME

Ecole : Saint Michel Annecy
Voie : SIO 1ère

Rédigé le : 10/04/2025

Sommaire :

Documentation Technique – BDD Musée

1. Introduction

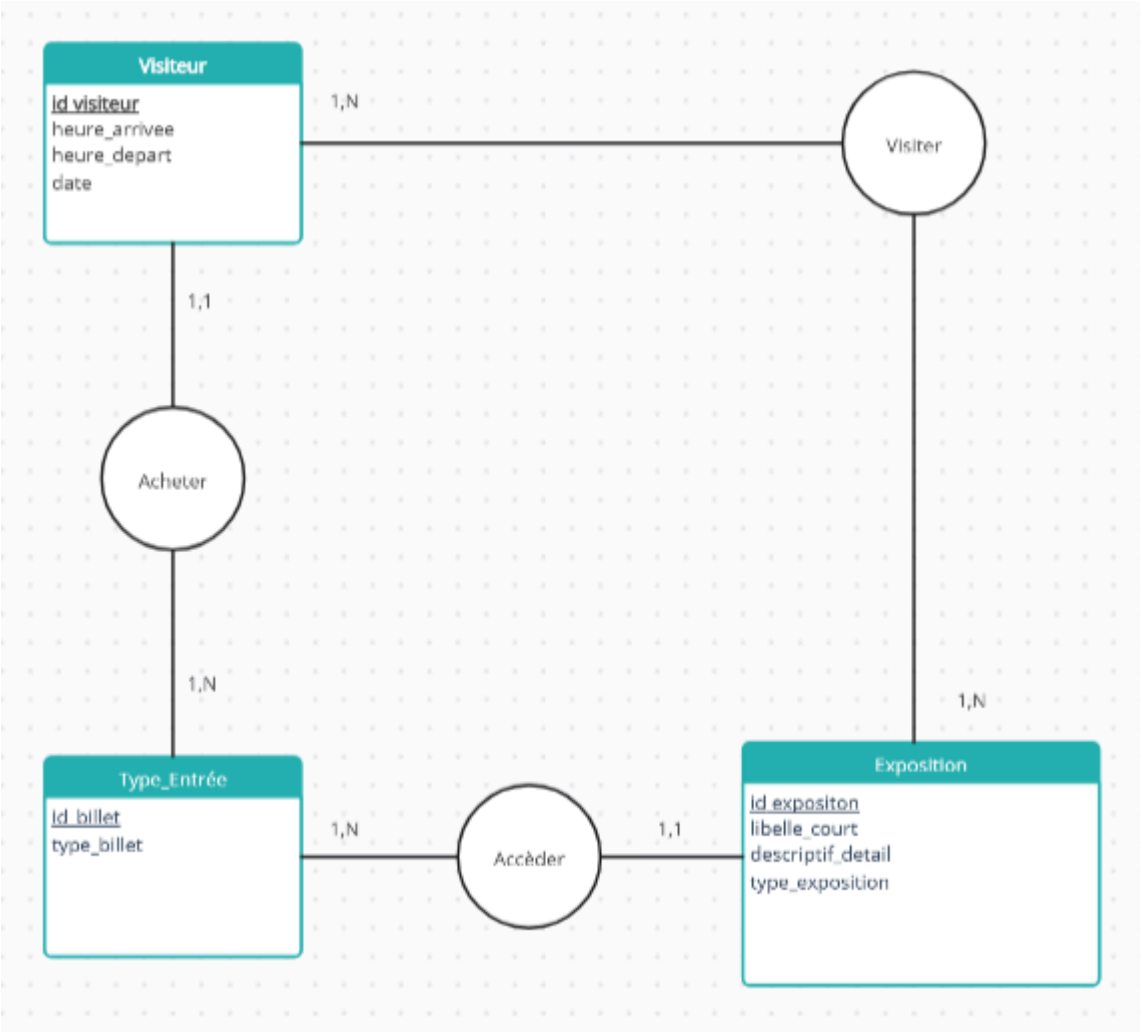
Objectif :
La base de données du musée permet de gérer :

- Les entrées/sorties des visiteurs (avec limite de 50 simultanés).
- Les expositions (permanentes/temporaires) et leurs dates.
- Les statistiques pour analyse financière.

2. Architecture Technique

2.1 Schéma de la Base de Données :

- Diagramme Relationnel (MCD) :



Tables Principales :

➢ `Visiteur` : Heures d'arrivée/départ, type d'entrée.
➢ `Exposition` : Libellé, type, dates de début/fin.
➢ `Type_Entree` : Catégorie de billet (Permanent/Temporaire/Les deux).
➢ `Utilisateur` : Comptes administrateurs (gestion des droits).

2.2 Technologies Utilisées :

- **SGBD** : MySQL
- **Outils** :
  - ➢ MySQL Workbench (modélisation).
  - ➢ PHPMyAdmin (administration).

3. Installation et Configuration

Prérequis :

- Serveur MySQL.
- Droits d'administration pour créer la BDD.

Instructions :

1. Exécutez le script `BDD Musée.sql` pour créer la structure.
2. Chargez les données initiales via `Exemple Insertion BDD Musée.sql`.
3. Créez les utilisateurs administrateurs avec `Création Utilisateurs Admin.sql`.

4. Utilisation

4.1 Fonctionnalités Clés :

- **Gestion des Visiteurs** :
  - Enregistrement automatique des entrées/sorties.
  - Blocage à 50 visiteurs simultanés.
- **Billetterie** :
  - Attribution des billets selon le type d'exposition.

4.2 Exemples de Requêtes SQL :

- **Statistiques Journalières** : `SELECT Type_Entree, COUNT(*) FROM Visiteur`

```
GROUP BY Type_Entree;
```

- **Vérification de la Capacité** : `SELECT COUNT(*) FROM Visiteur WHERE Heure_Depart IS NULL;`

5. Détails Techniques

5.1 Contraintes et Règles Métiers :

- **Limite de Visiteurs** :
  - Contrôle via trigger ou application (ex : message d'attente si 50 visiteurs présents).
- **Dates des Expositions** :
  - Temporaires : `Date_Debut` < `Date_Fin`.
  - Permanentes : `Date_Fin = '9999-12-31'`.

5.2 Gestion des Utilisateurs :

- **Rôles** :
  - `Admin` : Accès complet à la BDD.
  - `Utilisateur` : Droits limités (ex : lecture seule).

Documentation Technique – Code site web

1. Vue d'ensemble

Ce système est une application web PHP pour la gestion des visiteurs d'un musée.
Il comprend :

➢ Un tableau de bord administratif
➢ Un système d'enregistrement des visiteurs
➢ Des fonctionnalités de statistiques
➢ Une interface de gestion des utilisateurs

2. Architecture Technique

2.1 Fichiers Principaux :

- **fonctions_AdminPanel.php :** Contient toutes les fonctions back-end
- **dashboard.php :** Interface utilisateur principale
- **chart1.js / chart2.js :** Scripts pour les graphiques

2.2 Technologies Utilisées :

- PHP 8.4
- MySQL
- HTML, CSS
- JavaScript (Chart.js pour les graphiques)
- Icônes (Ionicons, Boxicons)

2.3 Description des dossiers et fichiers :

- **visiteurs/**
  - **img/:** Contient des images de tickets (ticket1.png, ticket2.png, ticket3.png).
  - **ticket1.pdf, ticket2.pdf, ticket3.pdf:** Fichiers PDF des tickets.
  - **ticket1.php, ticket2.php, ticket3.php:** Fichiers PHP associés aux tickets.
  - **index.php:** Page principale pour les visiteurs.
  - **vanilla-tilt.js:** Script JavaScript pour des effets visuels.

- **AdminPanel/**
  - **img/:** Contient des scripts JavaScript pour les graphiques (chart1.js, chart2.js).
  - **dashboard.php:** Page de tableau de bord pour l'administrateur.
  - **fonctions_AdminPanel.php:** Fichier PHP contenant des fonctions pour le panneau d'administration.

- **BDD/**
  - **BDD Musée.sql:** Script SQL pour la base de données du musée.
  - **Création Utilisateurs Admin.sql:** Script SQL pour la création des utilisateurs administrateurs.
  - **Exemple Insertion BDD Musée.sql:** Exemple d'insertion de données dans la base de données du musée.

- **login/**
  - **login.php:** Page de connexion.
  - **oublie.php:** Page pour la récupération de mot de passe oublié.
- **styles/**
  - **style.css:** Fichier CSS principal pour le style général.
  - **style_Dashboard.css:** Fichier CSS pour le style du tableau de bord.
  - **styleLogin.css:** Fichier CSS pour le style de la page de connexion.
  - **styleTickets.css:** Fichier CSS pour le style des tickets.

3. Fonctionnalités Principales

3.1 Gestion des Visiteurs :

- **Enregistrement :** Nom, prénom, type d'exposition(1,2 et 3)
    - *Exposition 1 = Entrée pour l'exposition permanente.*
    - *Exposition 2 = Entrée pour l'exposition temporaire.*
    - *Exposition 3 = Entrée pour les deux expositions.*

- **Départ :** Marquer le départ individuel ou de tous les visiteurs
- **Recherche :** Par nom, prénom, ID ou type d'exposition
- **Tri :** Par nom (A-Z/Z-A), date ou ID
- **Filtre :** Visiteurs actuels ou tous les visiteurs

3.2 Statistiques :

- **Graphique à barres :** Répartition par type d'entrée (filtrable par date)
- **Graphique linéaire :** Nombre de visiteurs sur 7 jours
- **Indicateurs :**
    - Visiteurs aujourd'hui
    - Visiteurs actuels (avec limite de 50)
    - Total des visiteurs

3.3 Gestion des Utilisateurs :

- Création de comptes avec différents rôles (Utilisateur/Admin)

4. Documentation des Fonctions

4.1 Connexion à la Base de Données :

Fonction « LoadBD » :

- **Description :** Établit une connexion PDO à la base de données
- **Paramètres :** Aucun
- **Retour :** Objet PDO

```php
function LoadBD()
{
    $host = "localhost";
    $dbname = "utilisateurs_db";
    $username = "root";
    $password = "";

    try {
        $pdo = new PDO( dsn: "mysql:host=$host;dbname=$dbname;charset=utf8", $username, $password);
        $pdo->setAttribute( attribute: PDO::ATTR_ERRMODE, value: PDO::ERRMODE_EXCEPTION);
        return $pdo;
    } catch (PDOException $e) {
        die("Erreur de connexion : " . $e->getMessage());
    }
}
```

4.2 Gestion des Visiteurs :

Fonction « LoadVisiteur » :

function LoadVisiteur($ordre, $date, $ID, $total, $searchField = null, $searchValue = null)

- **Description :** Récupère la liste des visiteurs avec filtres et tris
- **Paramètres :**
  - $ordre : Tri ASC/DESC
  - $date : Tri par date si true
  - $ID : Tri par ID si true
  - $total : Affiche tous les visiteurs si true, seulement les actuels sinon
  - $searchField : Champ de recherche
  - $searchValue : Valeur de recherche
- **Retour :** Tableau des visiteurs

```php
function LoadVisiteur($ordre, $date, $ID, $total, $searchField = null, $searchValue = null) {
    $pdo = LoadBD();

    $query = "SELECT * FROM visiteur";
    $conditions = [];
    $params = [];

    // Heure_Depart = 0
    if ($total === 'true') {
        $conditions[] = "Heure_Depart = 0";
    }

    // Recherche personnalisée
    $allowedFields = ['Nom', 'Prenom', 'ID_Visiteur', 'ID_Type_Entree'];
    if (!empty($searchField) && !empty($searchValue) && in_array($searchField, $allowedFields)) {
        if ($searchField === 'ID_Visiteur') {
            $conditions[] = "$searchField = ?";
            $params[] = $searchValue; // pas de %
        } else {
            $conditions[] = "$searchField LIKE ?";
            $params[] = "%$searchValue%";
        }
    }

    // Ajout de WHERE si nécessaire
    if (!empty($conditions)) {
        $query .= " WHERE " . implode( separator: " AND ", $conditions);
    }
```

```php
    // Ordre
    if ($ID === 'true') {
        $orderBy = 'ID_Visiteur';
    } elseif ($date === 'true') {
        $orderBy = 'Date_Visite';
    } else {
        $orderBy = 'Nom';
    }

    $orderDirection = ($ordre === 'true') ? 'ASC' : 'DESC';
    $query .= " ORDER BY $orderBy $orderDirection";

    // Préparation et exécution
    $stmt = $pdo->prepare($query);
    $stmt->execute($params);

    return $stmt->fetchAll();
}
```

Fonction « visiteur » :

function visiteur($nom, $prenom, $Arrivee, $Depart, $date_envoi, $exposition)

- **Description :** Ajoute un nouveau visiteur
- **Contrôle :** Vérifie qu'il y a moins de 50 visiteurs actuels

```php
function visiteur($nom, $prenom, $Arrivee, $Depart, $date_envoi, $exposition)
{
    $pdo = LoadBD();
    $NbVisiteurs=statsActu();

    if ($NbVisiteurs<50){
        $sql = " INSERT INTO Visiteur (Nom,Prenom,Heure_Arrivee, Heure_Depart, Date_Visite, ID_Type_Entree
            VALUES (:nom, :prenom, :Arrivee, :Depart, :date_envoi, :exposition)";
        $stmt = $pdo->prepare($sql);
        $stmt->execute([
            ':nom' => $nom,
            ':prenom' => $prenom,
            ':Arrivee' => $Arrivee,
            ':Depart' => $Depart,
            ':date_envoi' => $date_envoi,
            ':exposition' => $exposition
        ]);
    }else{
        echo "Erreur";
    }
}
```

Fonction « depart » :

function depart($valeurBouton, $heure)

- **Description :** Marque le départ d'un visiteur spécifique

```php
function depart($valeurBouton, $heure)
{
    $pdo = LoadBD();

    $sql = "UPDATE visiteur SET Heure_Depart = :heure WHERE ID_Visiteur = :id";
    $test = $pdo->prepare($sql);
    $test->execute([
        ':id' => $valeurBouton,
        ':heure' => $heure
    ]);
}
```

Fonction « departAll » : **function departAll($heure)**
• **Description :** Marque le départ de tous les visiteurs actuels

```php
function departAll($heure)
{
    $pdo = LoadBD();

    $sql = "UPDATE visiteur SET Heure_Depart = :heure WHERE Heure_Depart = 0";
    $test = $pdo->prepare($sql);
    $test->execute([
        ':heure' => $heure
    ]);
}
```

4.3 Statistiques :

Fonction « statsJour » : **function statsJour()**
• **Retour :** Nombre de visiteurs aujourd'hui

```php
function statsJour()
{
    $pdo = LoadBD();
    $query = "SELECT COUNT(*) FROM visiteur WHERE DATE(Date_Visite) = CURDATE();";

    $Result = $pdo->prepare($query);
    $Result->execute();

    $stats = $Result->fetchColumn();

    return (int)$stats;
}
```

Fonction « statsActu » : **function statsActu()**
• **Retour :** Nombre de visiteurs actuellement présents

```php
function statsActu()
{
    $pdo = LoadBD();
    $query = "SELECT COUNT(*) FROM visiteur WHERE Heure_Depart = 0;";

    $Result = $pdo->prepare($query);
    $Result->execute();

    $stats = $Result->fetchColumn();

    return (int)$stats;
}
```

Fonction « visiteur » : **function statsTotal()**
• **Retour :** Nombre total de visiteurs enregistrés

```php
function statsTotal()
{
    $pdo = LoadBD();
    $query = "SELECT COUNT(*) FROM visiteur;";

    $Result = $pdo->prepare($query);
    $Result->execute();

    $stats = $Result->fetchColumn();

    return (int)$stats;
}
```

Fonction « LoadDiagBar » :

function LoadDiagBar($dateDebut, $dateFin)

- **Description :** Données pour le graphique à barres (par type d'entrée)
- **Paramètres :** Dates optionnelles pour filtrer
- **Retour :** Tableau de données

```php
function LoadDiagBar($dateDebut, $dateFin) {
    $pdo = LoadBD();
    $barChartData = [];

    // Nettoyage : convertir '' en null
    $dateDebut = !empty($dateDebut) ? $dateDebut : null;
    $dateFin = !empty($dateFin) ? $dateFin : null;

    $query = "SELECT COUNT(*) FROM visiteur WHERE ID_Type_Entree = :typeEntree";

    // Ajout de la condition si les deux dates sont valides
    if (!empty($dateDebut) && !empty($dateFin)) {
        $query .= " AND Date_Visite >= :dateDebut AND Date_Visite <= :dateFin";
    }

    for ($type = 1; $type <= 3; $type++) {
        $stmt = $pdo->prepare($query);
        $stmt->bindValue( param: ':typeEntree', $type, type: PDO::PARAM_INT);

        if (!empty($dateDebut) && !empty($dateFin)) {
            $stmt->bindValue( param: ':dateDebut', $dateDebut);
            $stmt->bindValue( param: ':dateFin', $dateFin);
        }

        $stmt->execute();
        $barChartData[] = (int)$stmt->fetchColumn();
    }

    return $barChartData;
}
```

Fonction « LoadDiagLigne » :

function LoadDiagLigne()

- **Description :** Données pour le graphique linéaire (6 derniers jours + jour actuel)
- **Retour :** Tableau de données

```php
function LoadDiagLigne() {
    $pdo = LoadBD();
    $currentDate = new DateTime();
    $dates = [];
    $lineChartData = [];

    // Ajouter le jour actuel
    $dates[] = $currentDate->format( format: 'Y-m-d');

    // Ajouter les six jours précédents
    for ($i = 1; $i <= 6; $i++) {
        $previousDate = clone $currentDate;
        $previousDate->sub(new DateInterval( duration: "P{$i}D"));
        $dates[] = $previousDate->format( format: 'Y-m-d');
    }

    $dates = array_reverse($dates);

    foreach ($dates as $date) {
        $query = "SELECT COUNT(*) FROM visiteur WHERE Date_Visite = :date";
        $Result = $pdo->prepare($query);
        $Result->execute([':date' => $date]);
        $Nb = $Result->fetchColumn();
        array_push( &array: $lineChartData, $Nb);
    }

    return $lineChartData;
}
```

Fonction « LoadDiagLigneName » :

function LoadDiagLigneName()

- **Description :** Noms des jours pour l'axe X du graphique linéaire
- **Retour :** Tableau des noms de jours en français

```php
function LoadDiagLigneName(){
    $currentDate = new DateTime();
    $dates = [];

    // jour actuel
    $dates[] = clone $currentDate;

    // Ajouter les six jours précédents
    for ($i = 1; $i <= 6; $i++) {
        $previousDate = clone $currentDate;
        $previousDate->sub(new DateInterval( duration: "P{$i}D"));
        $dates[] = $previousDate;
    }

    $nomsDesJours = [];

    // Formater les noms des jours en français
    foreach ($dates as $date) {
        $formatter = new IntlDateFormatter(
            locale: 'fr_FR',
            dateType: IntlDateFormatter::FULL,
            timeType: IntlDateFormatter::NONE,
            timezone: 'Europe/Paris',
            calendar: IntlDateFormatter::GREGORIAN,
            pattern: 'EEEE'
        );
        $nomsDesJours[] = $formatter->format($date);
    }

    return array_reverse($nomsDesJours);
}
```

4.4 Gestion des Utilisateurs :

Fonction « visiteur » : **function adminPanel($user, $mdp, $role)**
• **Description :** Ajoute un nouvel utilisateur • **Paramètres :** Prend l'identifiant, le mot de passe, et le rôle (Administrateur, ou invité)

```php
function adminPanel($user, $mdp, $role)
{
    $pdo = LoadBD();

    $sql = "INSERT IGNORE INTO Utilisateurs (Pseudo, MDP, salut) VALUES (:user, :mdp, :role)";
    $stmt = $pdo->prepare($sql);
    $stmt->execute([
        ':user' => $user,
        ':mdp' => $mdp,
        ':role' => $role
    ]);
}
```

5. Interface Utilisateur (dashboard.php)

5.1 Structure :

- **Navigation :** Barre latérale avec 3 sections principales et 1 option de déconnexion
- **Sections :**
    1. Accueil : Formulaires d'ajout
    2. Statistiques : Graphiques et indicateurs
    3. Visiteurs : Liste et gestion

5.2 Fonctionnalités Front-end :

➔ **Navigation :** Changement d'état actif au scroll/clic
➔ **Boutons de tri :** Changement d'état visuel
➔ **Graphiques :** Affichage dynamique via Chart.js
➔ **Ancres :** Navigation fluide entre sections

6. Sécurité

- **Validation :** Tous les champs de formulaire sont requis
- **PDO :** Utilisation de requêtes préparées
- **Contrôle d'accès :** Limite de 50 visiteurs simultanés

Site protéger contre les injections SQL.

7. Points d'Amélioration

Sécurité :

- Gestion des sessions utilisateur
- Hash des mots de passe

Fonctionnalités :

- Pagination pour la liste des visiteurs
- Export des données

Interface :

- Messages de confirmation/erreur
- Design responsive pour le panel de gestion

8. Installation

- Configurer la base de données (non détaillée)
- Modifier les paramètres de connexion dans LoadBD()
- Déployer les fichiers sur un serveur web avec PHP

9. Dépendances

- Chart.js (via CDN)
- Ionicons (via CDN)
- Boxicons (via CDN)

Annexes

- **Scripts SQL** :
    - `BDD Musée.sql`
    - `Exemple Insertion BDD Musée.sql`.
- **Références** :
    - Documentation MySQL : https://dev.mysql.com/doc/.
    - Documentation Php : https://www.php.net/manual/fr/index.php
    - Documentation Js : https://devdocs.io/javascript/
    - Documentation Css : https://www.w3schools.com/CSSref/index.php
    - Documentation HTML : https://developer.mozilla.org/en-US/docs/Web/HTML
    - M.PHILLIPS