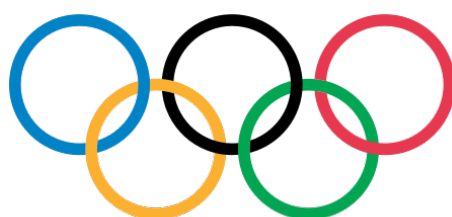


CAHIER DES CHARGES TECHNIQUES

SITE WEB JO2024



PARIS 2024





SOMMAIRE

- [1. Contexte du projet](#)
 - [1.1. Présentation du projet](#)
 - [1.2. Date de rendu du projet](#)
- [2. Besoins fonctionnels](#)
- [3. Ressources nécessaires à la réalisation du projet](#)
 - [3.1. Ressources matérielles](#)
 - [3.2. Ressources logicielles](#)
- [4. Gestion du projet](#)
- [5. Conception du projet](#)
 - [5.1. Le front-end](#)
 - [5.1.1. Wireframes](#)
 - [5.1.2. Maquettes](#)
 - [5.1.3. Arborescences](#)
 - [5.2. Le back-end](#)
 - [5.2.1. Diagramme de cas d'utilisation](#)
 - [5.2.2. Diagramme d'activités](#)
 - [5.2.3. Modèles Conceptuel de Données \(MCD\)](#)
 - [5.2.4. Modèle Logique de Données \(MLD\)](#)
 - [5.2.5. Modèle Physique de Données \(MPD\)](#)
- [6. Technologies utilisées](#)
 - [6.1. Langages de développement Web](#)
 - [6.2. Base de données](#)
- [7. Sécurité](#)
 - [7.1. Login et protection des pages administrateurs](#)
 - [7.2. Cryptage des mots de passe avec Bcrypt](#)
 - [7.3. Protection contre les attaques XSS \(Cross-Site Scripting\)](#)
 - [7.4. Protection contre les injections SQL](#)

1. Contexte du projet

1.1. Présentation du projet

Votre agence web a été sélectionnée par le comité d'organisation des jeux olympiques de Paris 2024 pour développer une application web permettant aux organisateurs, aux médias et aux spectateurs de consulter des informations sur les sports, les calendriers des épreuves et les résultats des JO 2024.

Votre équipe et vous-même avez pour mission de proposer une solution qui répondra à la demande du client.

1.2. Date de rendu du projet

Le projet doit être rendu au plus tard le 22 mars 2024.

2. Besoins fonctionnels

Le site web devra avoir une partie accessible au public et une partie privée permettant de gérer les données.

Les données seront stockées dans une base de données relationnelle pour faciliter la gestion et la mise à jour des informations. Ces données peuvent être gérées directement via le site web à travers un espace administrateur.

3. Ressources nécessaires à la réalisation du projet

3.1. Ressources matérielles

Les ressources matérielles nécessaires à la réalisation du projet sont :

- Ordinateur portable

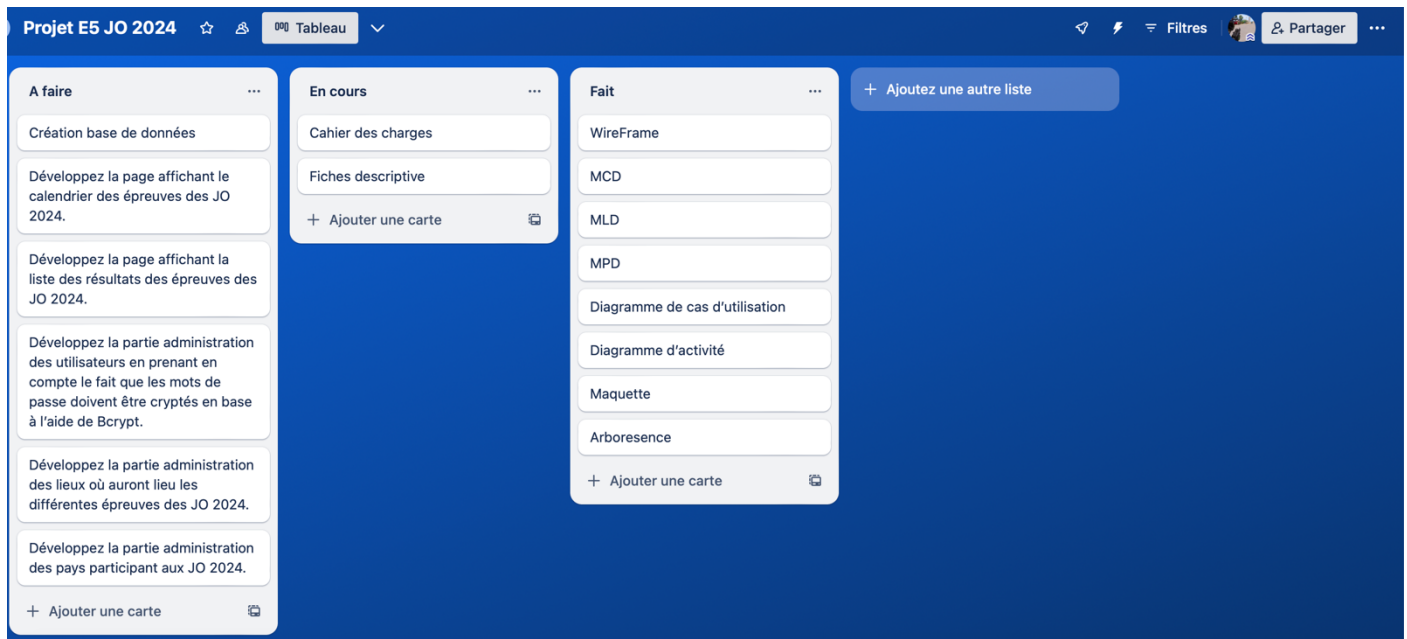
3.2. Ressources logicielles

Les ressources logicielles nécessaires à la réalisation du projet sont :

- Visual Studio Code
- MAMP
- GitHub
- Trello

4. Gestion du projet

Pour réaliser le projet, nous utiliserons la méthode Agile Kanban. Nous utiliserons également l'outil de gestion de projet en ligne Trello.



Nous travaillons également sur GitHub, plateforme de développement collaboratif.

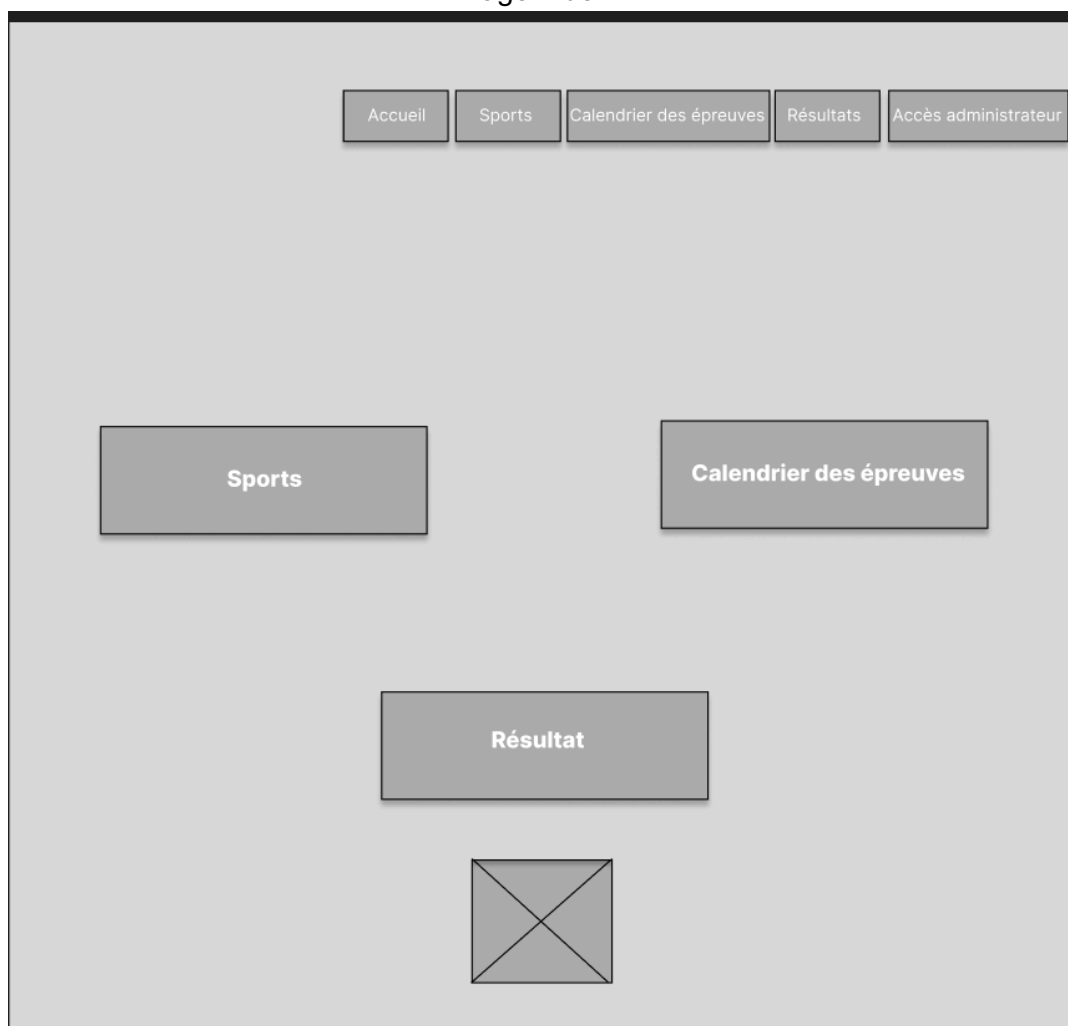
5. Conception du projet

5.1. Le front-end

5.1.1. Wireframes

Wireframes version ordinateur :

Page Index :



Page connexion :


Accueil	Sports	Calendrier des épreuves	Résultats	Accès administrateur
---------	--------	-------------------------	-----------	----------------------

Connexion

Login :

Mot de passe

Se connecter



Accueil Administration Gestion Sports Gestion Lieux Gestion Calendrier Gestion Pays Gestion Genres Gestion Athlètes Gestion Résultat Deconnexion

Ajouter un Sport

Nom du sport :

Ajouter le sport

Retour à la gestion des sports



Wireframes version responsive :

Page Index responsive :



Page connexion responsive :

Accueil

Sports

Calendrier des épreuves

Résultats

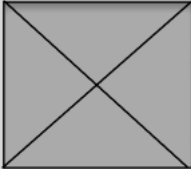
Accès administrateur

Connexion

Login :

Mot de passe

Se connecter



Page ajout de sport responsive :

The image displays a responsive web interface for adding a sport. It features a vertical menu of options and a form for adding a new sport.

Menu Options:

- Accueil Administration
- Gestion Sports
- Gestion Lieux
- Gestion Calendrier
- Gestion Pays
- Gestion Genres
- Gestion Athlètes
- Gestion Résultat
- Deconnexion

Ajouter un Sport

Nom du sport :

5.1.2. Maquettes

Maquettes version ordinateur :

Index :

[Accueil](#) [Sports](#) [Calendrier des épreuves](#) [Résultats](#) [Accès administrateur](#)

Sports

Calendrier des épreuves

Résultats



Connexion :

[Accueil](#) [Sports](#) [Calendrier des événements](#) [Résultats](#) [Accès administrateur](#)

Connexion

Login :

Mot de passe :

[Se connecter](#)



Ajout de sport :



Ajouter un Sport

Nom du Sport :

Ajouter le Sport

Retour à la gestion des sports



Maquettes version responsive :

Index :



Sports

Calendrier des épreuves

Résultats



Connexion :

Accueil

Sports

Calendrier des évènements

Résultats

Accès administrateur

Connexion

Login :

Mot de passe :

Se connecter



Ajout de sport :

Accueil Administration

Gestion Sports

Gestion Lieux

Gestion Calendrier

Gestion Pays

Gestion Genres

Gestion Athlètes

Gestion Résultats

Déconnexion

Ajouter un Sport

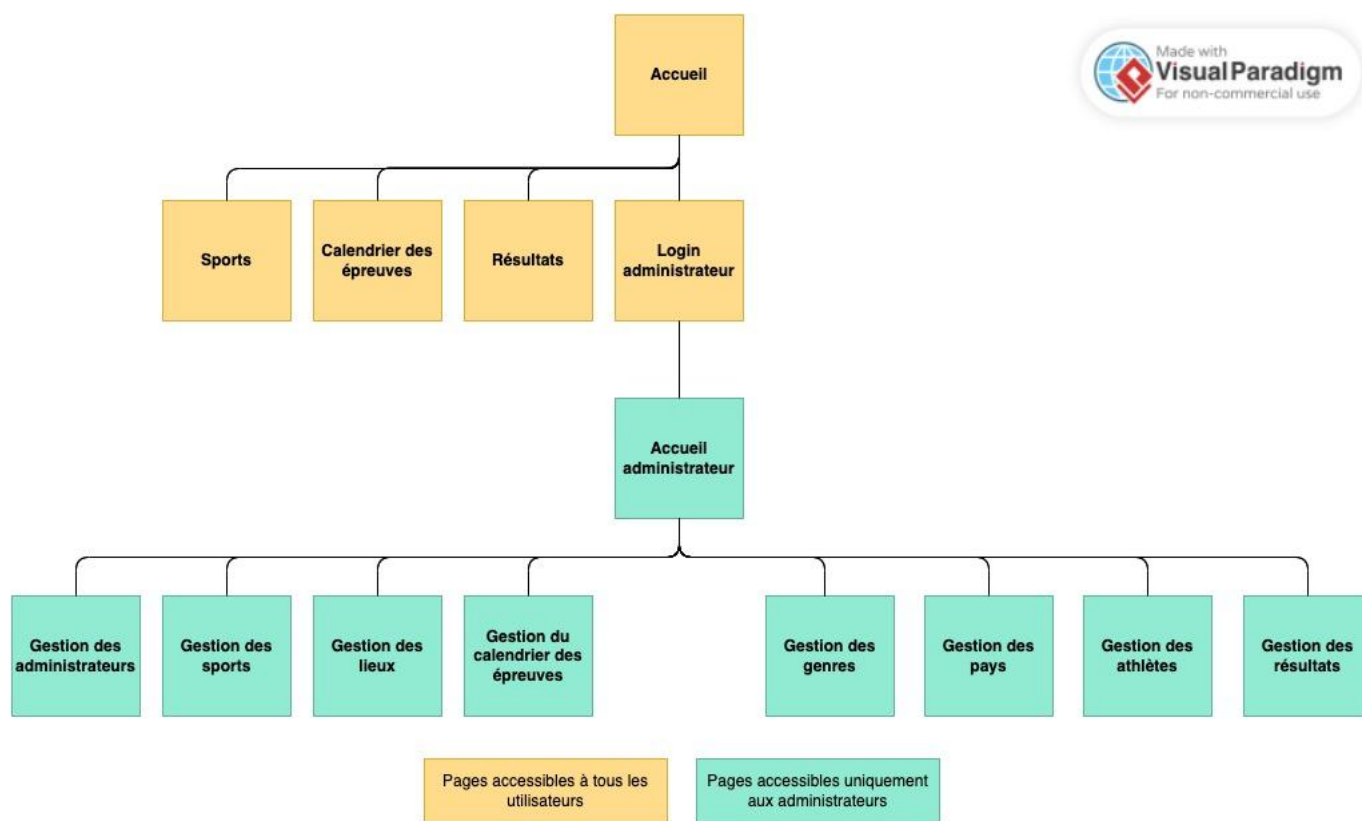
Nom du Sport :

Ajouter le Sport

Retour à la gestion des sports

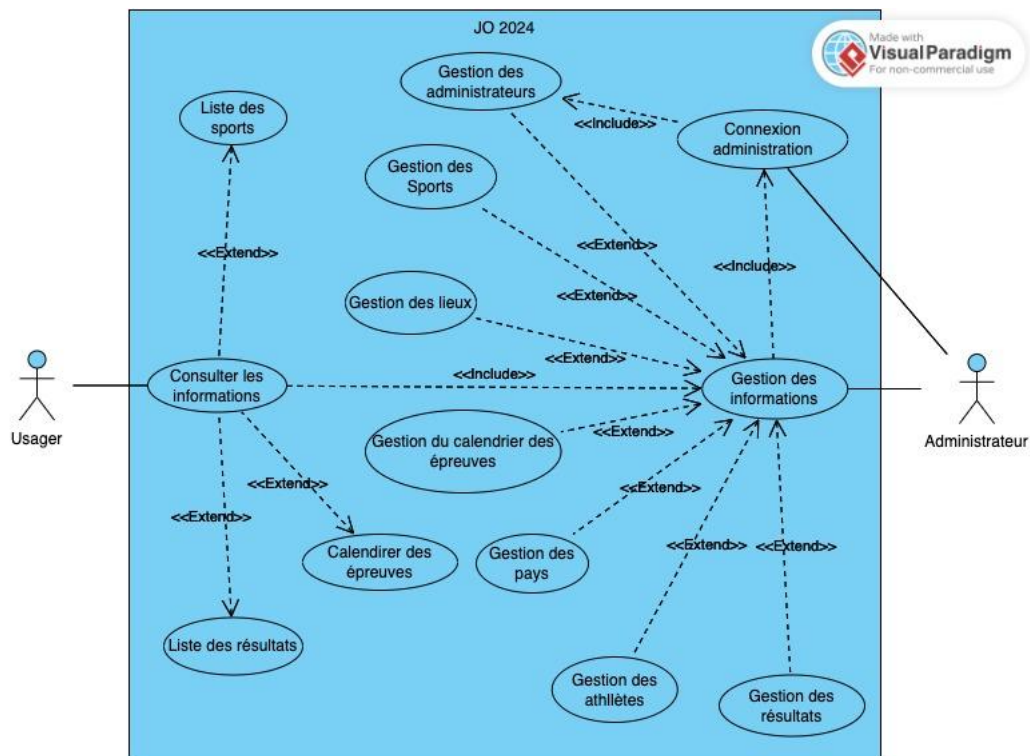


5.1.3. Arborescences

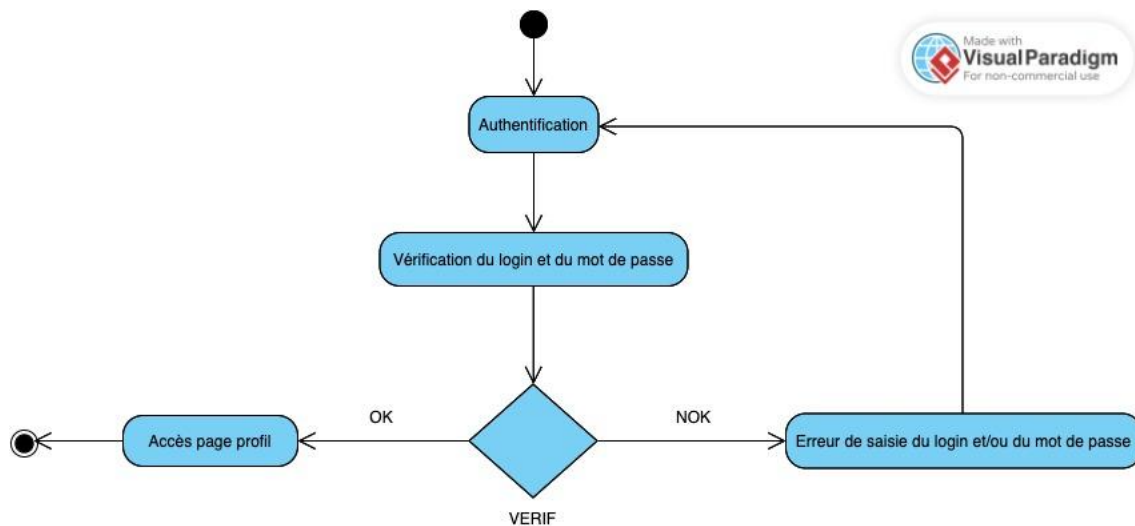


5.2. Le back-end

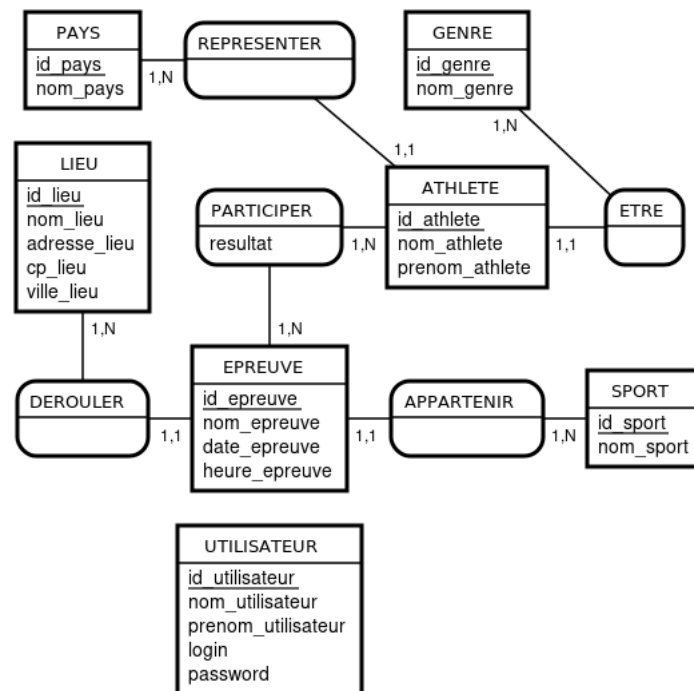
5.2.1. Diagramme de cas d'utilisation



5.2.2. Diagramme d'activités



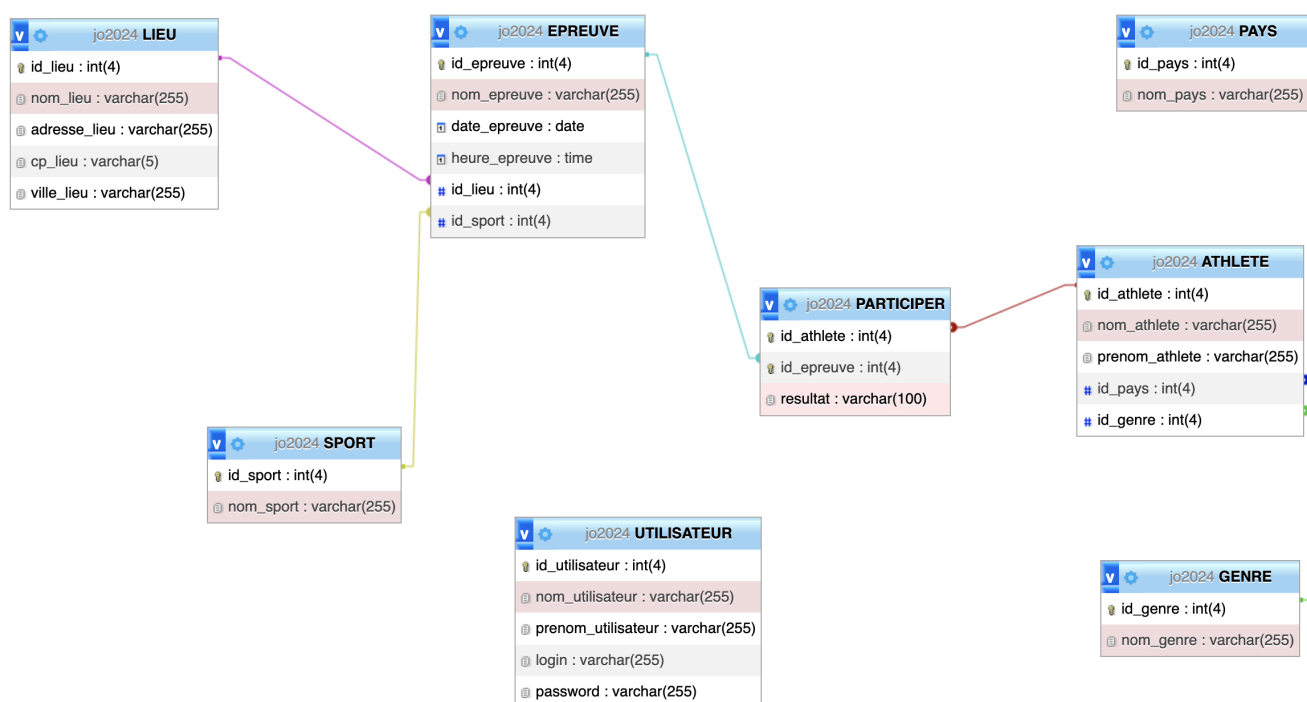
5.2.3. Modèles Conceptuel de Données (MCD)



5.2.4. Modèle Logique de Données (MLD)

Entité ou association	Libellé de l'attribut	Type
ATHLETE	id_athlete	INT(4)
/	nom_athlete	VARCHAR(255)
/	prenom_athlete	VARCHAR(255)
EPREUVE	id_epreuve	INT(4)
/	nom_epreuve	VARCHAR(255)
/	date_epreuve	DATE
/	heure_epreuve	TIME
GENRE	id_genre	INT(4)
/	nom_genre	VARCHAR(255)
LIEU	id_lieu	INT(4)
/	nom_lieu	VARCHAR(255)
/	adresse_lieu	VARCHAR(255)
/	cp_lieu	VARCHAR(5)
/	ville_lieu	VARCHAR(255)
PARTICIPER	resultat	VARCHAR(100)
PAYS	id_pays	INT(4)
/	nom_pays	VARCHAR(255)
SPORT	id_sport	INT(4)
/	nom_sport	VARCHAR(255)
UTILISATEUR	id_utilisateur	
	nom_utilisateur	
	prenom_utilisateur	
	login	
	mdp	

5.2.5. Modèle Physique de Données (MPD)



6. Technologies utilisées

6.1. Langages de développement Web

Les langages de développement web utilisé pour le projet sont :

- HTML
- CSS
- PHP
- JavaScript

6.2. Base de données

La base de données utilisé est MySQL

7. Sécurité

7.1. Login et protection des pages administrateurs

On récupère nos identifiants et mot de passe à l'aide de la méthode POST. Ensuite, on vérifie notre mot de passe ainsi que l'identifiant. Si les identifiants et le mot de passe sont corrects, on redirige vers la page administrateur et on lance une session. Si les accès ne sont pas corrects, on redirige vers l'index.

```
connexion.php x
connexion.php
1  <?php
2
3  error_reporting(E_ALL); ini_set("display_errors", 1);
4
5  session_start();
6
7  $username_registered = 'said';
8  $password_hash = '$2y$10$Lh1Sz1/mbdgcBUNyRfHL9ebA64kSyePIzNEN.11TbxXs.kwIn13nu';
9  $password = $_POST['password'];
10 $username = $_POST['username'];
11
12 if (password_verify($password, $password_hash) && $username_registered == $username) {
13     echo "Connexion réussie !";
14     header('Location: profile.php');
15 } else {
16     header('Location: index.php');
17 }
18
19 ?>
```

7.2. Cryptage des mots de passe avec Bcrypt

Bcrypt est un algorithme de hachage de mots de passe conçu pour être lent et résistant aux attaques par force brute. Il est particulièrement utilisé pour stocker de manière sécurisée les mots de passe dans les bases de données.

```
add-users.php X
pages > admin > admin-users > add-users.php
17 $mdp_utilisateur = filter_input(INPUT_POST, 'mdpUtilisateur', FILTER_SANITIZE_STRING);
18
19 // Vérifiez si le nom d'utilisateur est vide
20 if (empty($nom_utilisateur)) {
21     $_SESSION['error'] = "Le nom de l'utilisateur ne peut pas être vide.";
22     header("Location: add-users.php");
23     exit();
24 }
25
26 try {
27     // Vérifiez si le nom d'utilisateur existe déjà
28     $queryCheck = "SELECT id_utilisateur FROM UTILISATEUR WHERE nom_utilisateur = :nomUtilisateur";
29     $statementCheck = $connexion->prepare($queryCheck);
30     $statementCheck->bindParam(":nomUtilisateur", $nom_utilisateur, PDO::PARAM_STR);
31     $statementCheck->execute();
32
33     if ($statementCheck->rowCount() > 0) {
34         $_SESSION['error'] = "Le nom d'utilisateur existe déjà.";
35         header("Location: add-users.php");
36         exit();
37     } else {
38         // Hacher le mot de passe
39         $hashed_password = password_hash($mdp_utilisateur, PASSWORD_BCRYPT);
40
41         // Requête pour ajouter un utilisateur
42         $query = "INSERT INTO UTILISATEUR (nom_utilisateur, prenom_utilisateur, login, password) VALUES (:nomUtilisateur, :p";
43         $statement = $connexion->prepare($query);
44         $statement->bindParam(":nomUtilisateur", $nom_utilisateur, PDO::PARAM_STR);
45         $statement->bindParam(":prenomUtilisateur", $prenom_utilisateur, PDO::PARAM_STR);
46         $statement->bindParam(":loginUtilisateur", $login_utilisateur, PDO::PARAM_STR);
47         $statement->bindParam(":mdpUtilisateur", $hashed_password, PDO::PARAM_STR);
48
49         // Exécutez la requête
50         if ($statement->execute()) {
51             $_SESSION['success'] = "L'utilisateur a été ajouté avec succès.";
52             header("Location: manage-users.php");
53             exit();
54         } else {
```

7.3. Protection contre les attaques XSS (Cross-Site Scripting)

Les attaques XSS surviennent lorsqu'un attaquant injecte du code JavaScript malveillant dans les pages web consultées par d'autres utilisateurs. On utilise la fonction `htmlspecialchars()` de cette manière : même si l'utilisateur entre du code JavaScript malveillant dans le champ de saisie, celui-ci sera affiché comme du texte brut dans la page, sans être interprété comme du code exécutable par le navigateur.

7.4. Protection contre les injections SQL

Une injection SQL est une façon d'accéder et d'effectuer sans autorisation des opérations sur une base de données en injectant une requête SQL, la plupart du temps cette injection se fait via un formulaire. Afin de pallier à ce problème, on utilise PDO. Ce sont des requêtes préparées qui permettent d'éviter les injections SQL.