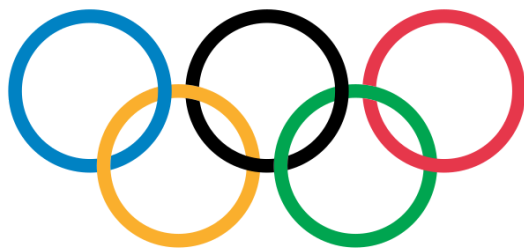


Cahier des charges JO2024



PARIS 2024



Cahier des charges techniques

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

REPRENDRE RÉPONSES MISSION 1

3.1. Ressources matérielles

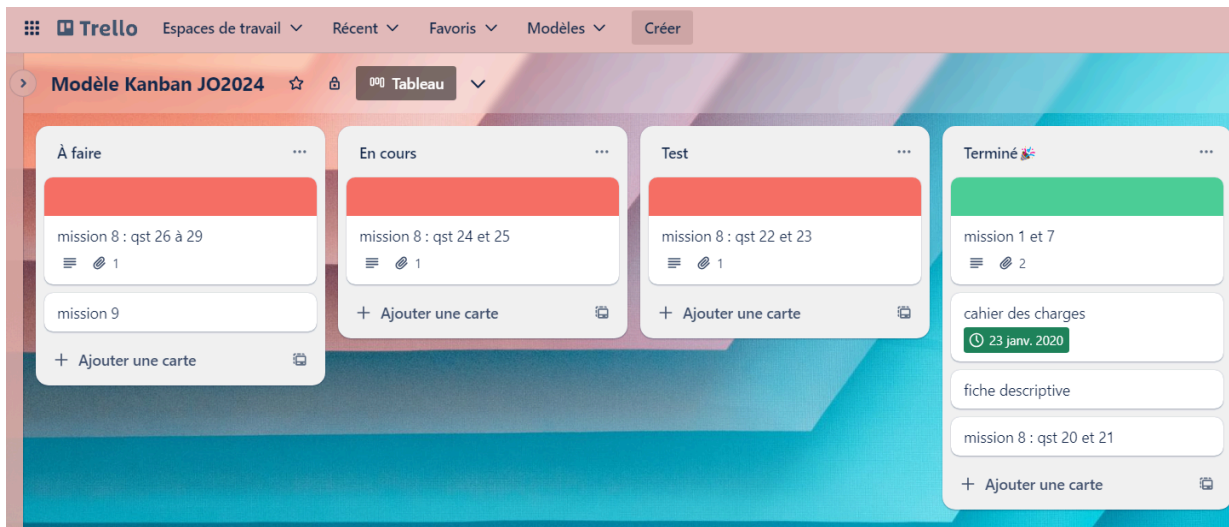
- Ordinateur
- Ecran
- Clavier
- Souris

3.2. Ressources logicielles

- IDE (Visual studio code)
- Trello
- Github
- MAMP
- Mocodo
- Visual Paradig
- MySQL

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.

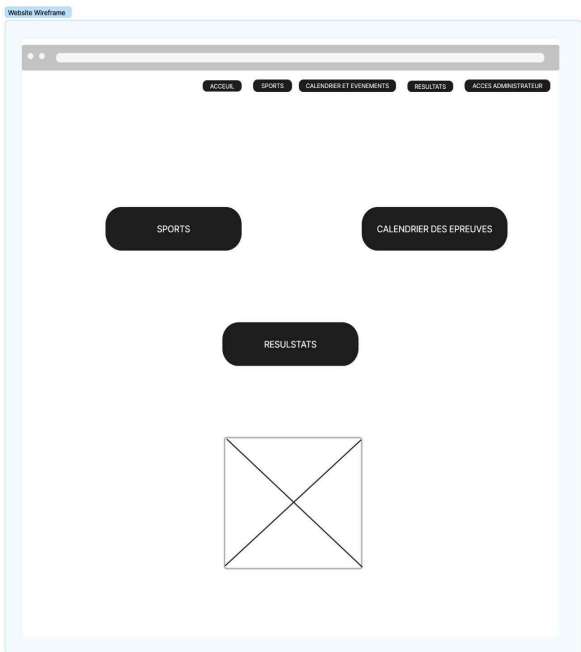
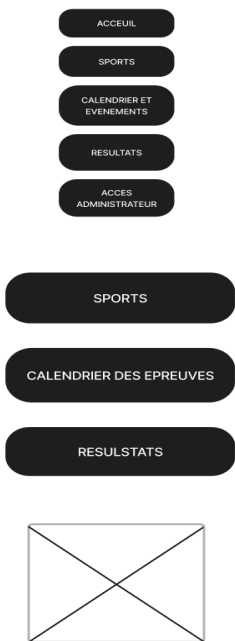


Nous travaillons également sur GitHub, plateforme de développement collaboratif.

5. Conception du projet

5.1. Le front-end

5.1.1. Wireframes

Ordinateur	Responsive
<p>Index :</p> 	<p>Index :</p> 

Ordinateur

Login :

Website Wireframe

ACCUEILSPORTSCALENDRIER ET EVENEMENTSRESULTATSACCES ADMINISTRATEUR

CONNEXION

Login :

Mot de passe :

SE CONNECTER

Responsive

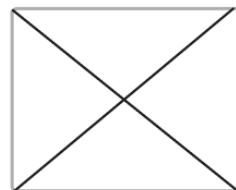
Login :

- ACCEUIL
- SPORTS
- CALENDRIER ET
EVENEMENTS
- RESULTATS
- ACCES
ADMINISTRATEUR

CONNEXION

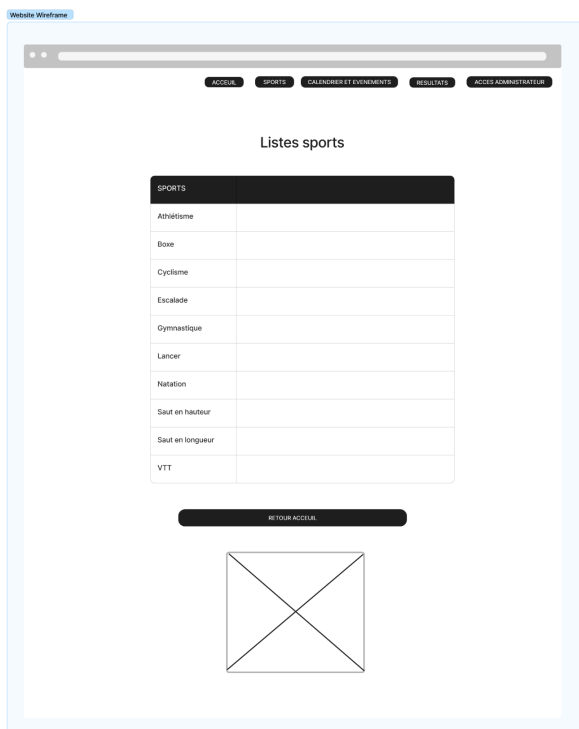
Login :

Mot de passe :



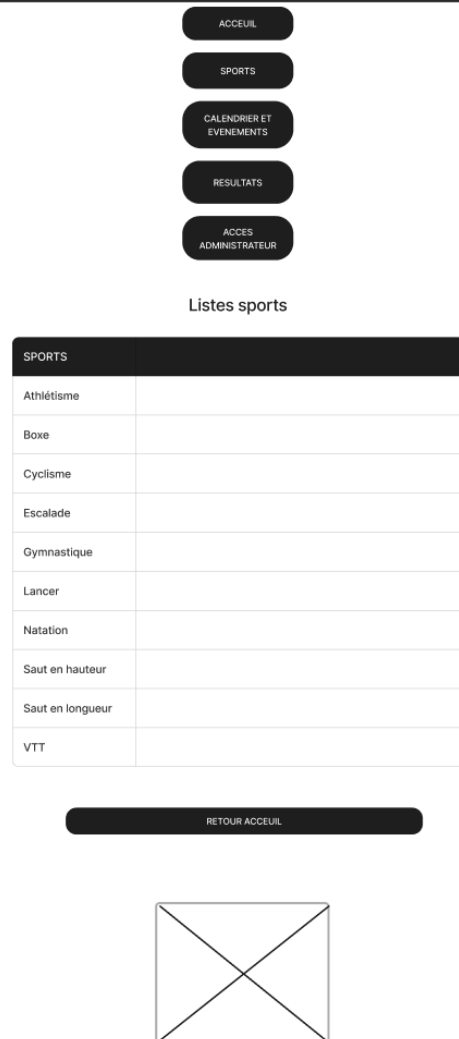
Ordinateur

Liste sports :





Responsive

Liste sports :



5.1.2. Maquettes

Ordinateur	Responsive
<p>index.php :</p> 	<p>index.php :</p> 

Ordinateur

login.php :

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

Connexion

Login :

Mot de passe :

[Se connecter](#)



Responsive

login.php :

[Accueil](#)

[Sports](#)

[Calendrier des événements](#)

[Résultats](#)

[Accès administrateur](#)

Connexion

Login :

Mot de passe :

[Se connecter](#)



Ordinateur

sports.php :

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

Liste des Sports

Sport
Athlétisme
Boxe
Cyclisme
Escalade
Gymnastique
Lancer
Natation
Saut en hauteur
Saut en longueur
VTT

[Retour Accueil](#)



Responsive

sports.php :

[Accueil](#)

[Sports](#)

[Calendrier des événements](#)

[Résultats](#)

[Accès administrateur](#)

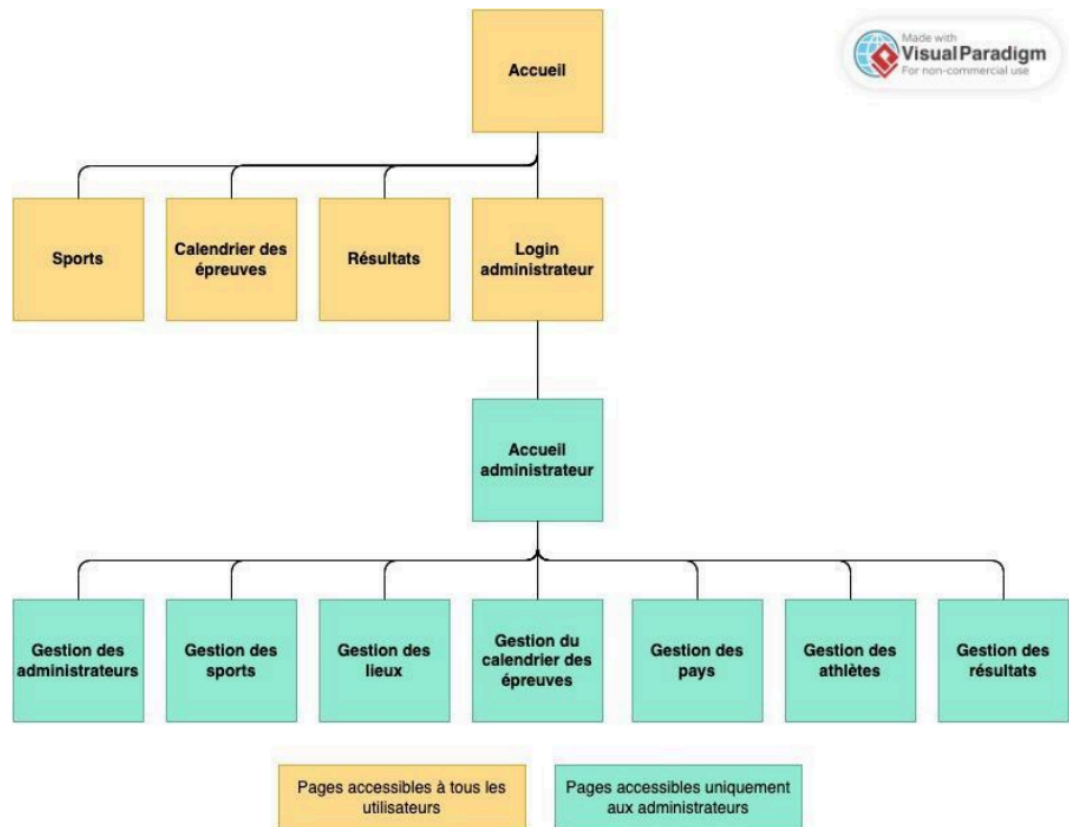
Liste des Sports

Sport
Athlétisme
Boxe
Cyclisme
Escalade
Gymnastique
Lancer
Natation
Saut en hauteur
Saut en longueur
VTT

[Retour Accueil](#)

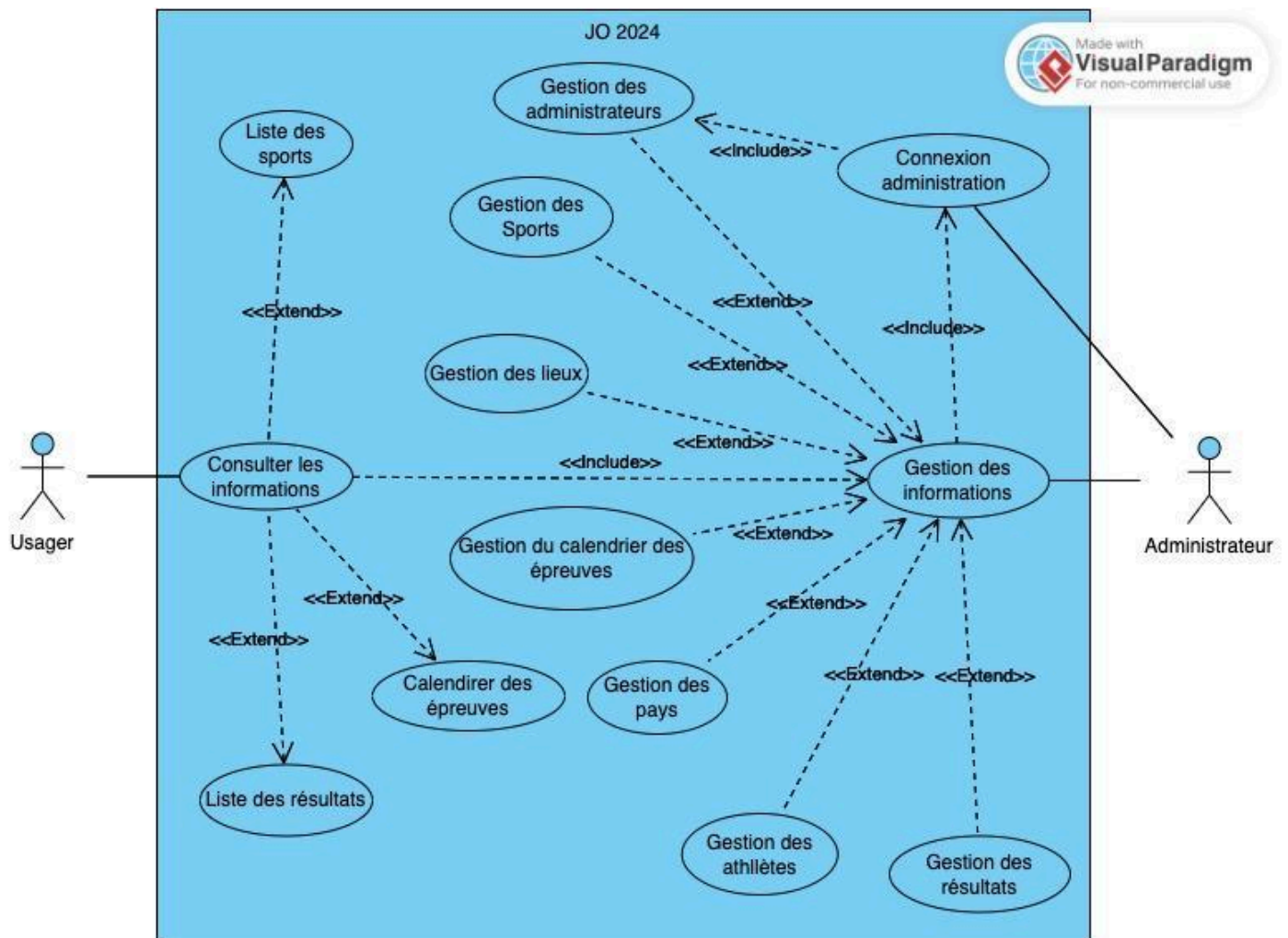


5.1.3. Arborescence



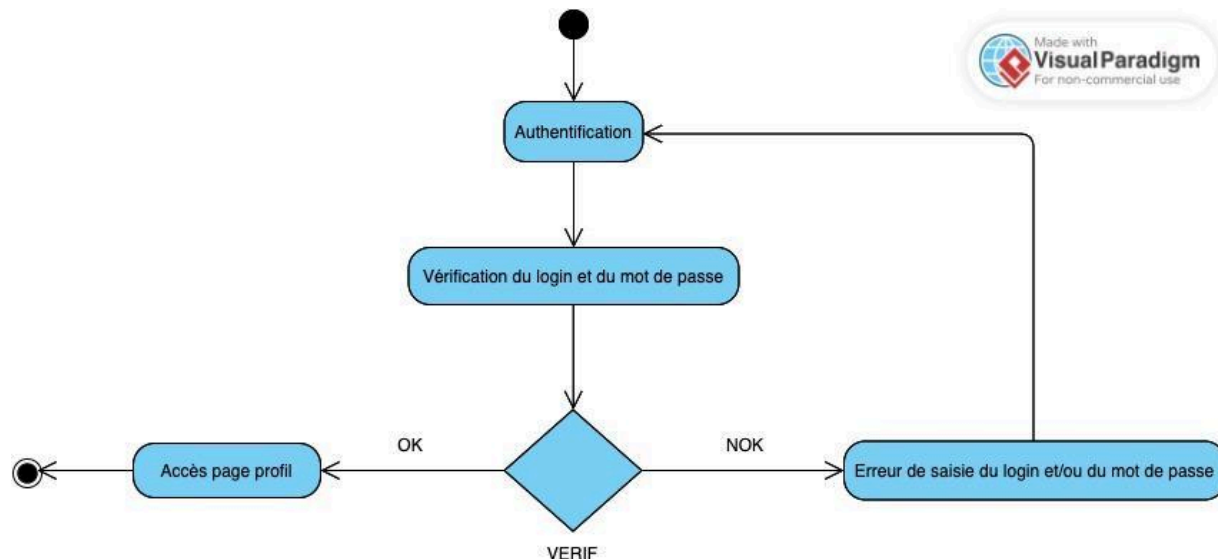
5.2. Le back-end

5.2.1. Diagramme de cas d'utilisation



5.2.2. Diagramme d'activités

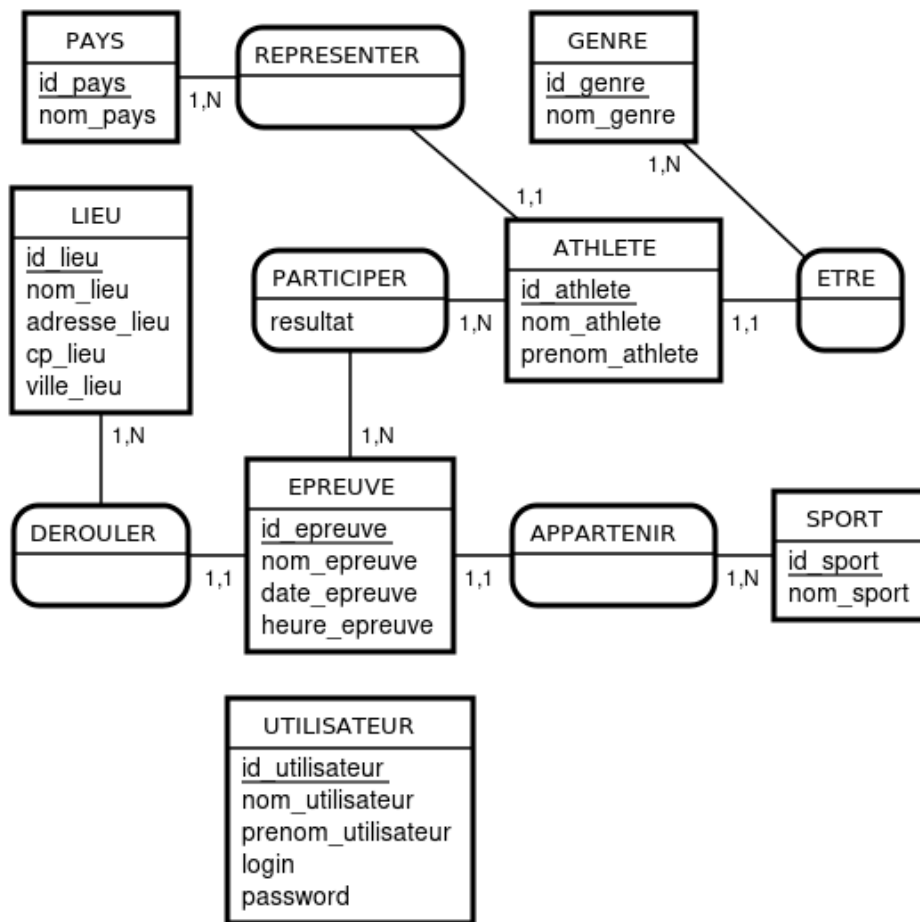
Ce diagramme présente la gestion des login.



5.2.3. Dictionnaire de données

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	INT(4)
/	nom_utilisateur	VARCHAR(255)
/	prenom_utilisateur	VARCHAR(255)
/	login	VARCHAR(255)
/	password	VARCHAR(255)

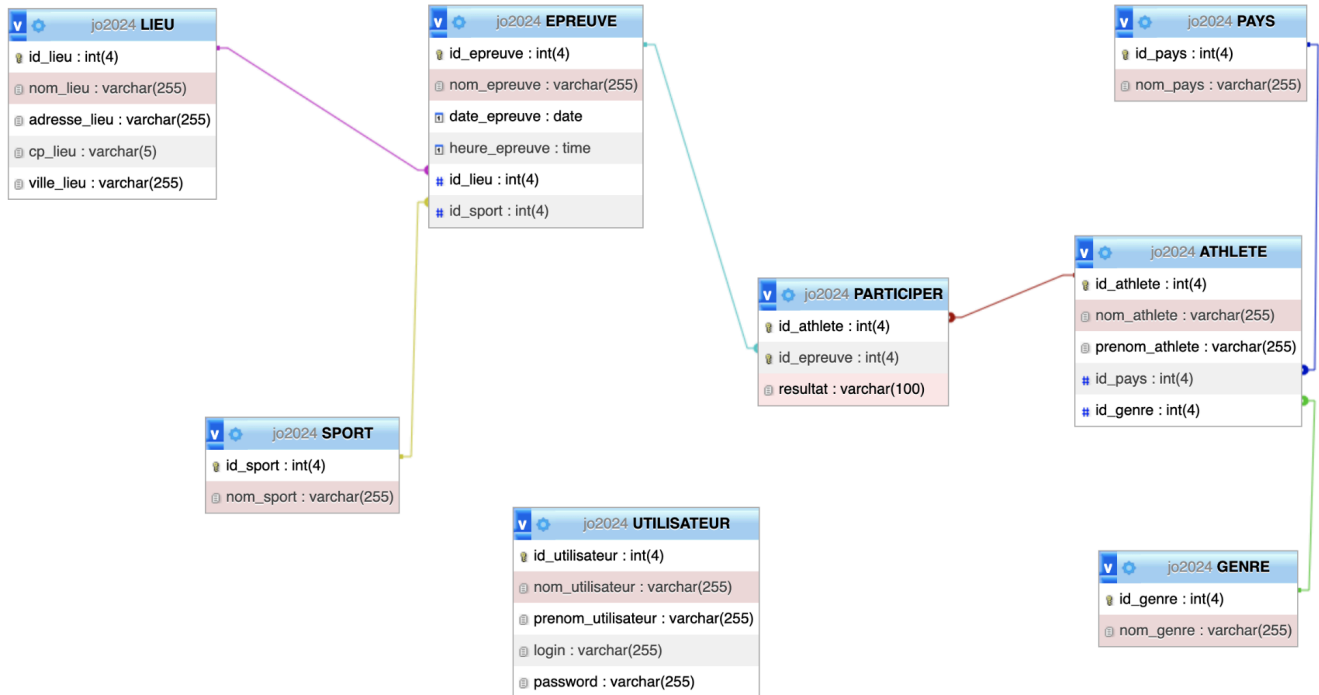
5.2.4. Modèles Conceptuel de Données (MCD)



5.2.5. Modèle Logique de Données (MLD)

- ATHLETE (id_athlete, nom_athlete, prenom_athlete, #id_pays, #id_genre)
- EPREUVE (id_epreuve, nom_epreuve, date_epreuve, heure_epreuve, #id_lieu, #id_sport)
- GENRE (id_genre, nom_genre)
- LIEU (id_lieu, nom_lieu, adresse_lieu, cp_lieu, ville_lieu)
- PARTICIPER (#id_athlete, #id_epreuve, resultat)
- PAYS (id_pays, nom_pays)
- SPORT (id_sport, nom_sport)
- UTILISATEUR (id_utilisateur, nom_utilisateur, prenom_utilisateur, login, password)

5.2.6. Modèle Physique de Données (MPD)



6. Technologies utilisées

6.1. Langages de développement Web

Les langages utilisés afin de développer l'application web sont :

- HTML 5
- CSS 3
- PHP 8.1

6.2. Base de données

- MySQL

7. Sécurité

7.1. Login et protection des pages administrateurs

Pour le login, nous envisageons la création d'un formulaire permettant à l'utilisateur de saisir son identifiant et son mot de passe. Ces informations seront utilisées pour une comparaison avec les données stockées dans la base de données afin de vérifier l'existence d'un compte correspondant à l'utilisateur en question.

```
30     <main>
31         <?php
32             // Affichage un message d'erreur si erreur lors de la tentative de connexion
33             session_start();
34             if (isset($_SESSION['error'])) {
35                 echo '<p style="color: red;">' . $_SESSION['error'] . '</p>';
36                 unset($_SESSION['error']);
37             }
38             // Afficher les erreurs en PHP
39             // (fonctionne à condition d'avoir activé l'option en local)
40             error_reporting(E_ALL);
41             ini_set("display_errors", 1);
42             ?>
43             <h1>Connexion</h1>
44             <form action="../database/auth.php" method="post">
45                 <label for="login">Login :</label>
46                 <input type="text" name="login" id="login" required><br><br>
47                 <label for="password">Mot de passe :</label>
48                 <input type="password" name="password" id="password" required><br><br>
49                 <input type="submit" value="Se connecter">
50             </form>
51     </main>
```

7.2. Cryptage des mots de passe avec Bcrypt

Bcrypt est une fonction de hachage utilisée pour sécuriser les mots de passe dans les applications web. Elle ralentit volontairement le processus de hachage pour rendre les attaques plus difficiles

```
39     // Hachage du mot de passe avec Bcrypt
40     $hashed_password = password_hash($mdp, PASSWORD_BCRYPT);
41
42     // Requête pour ajouter un utilisateur
43     $query = "INSERT INTO UTILISATEUR (nom_utilisateur, prenom_utilisateur, login, password) VALUES
44     (:nomUtilisateur, :prenomUtilisateur, :login, :hashed_password)";
45     $statement = $connexion->prepare($query); @var mixed $nomUtilisateur
46     $statement->bindParam(":nomUtilisateur", $nomUtilisateur, PDO::PARAM_STR);
47     $statement->bindParam(":prenomUtilisateur", $prenomUtilisateur, PDO::PARAM_STR);
48     $statement->bindParam(":login", $login, PDO::PARAM_STR);
49     $statement->bindParam(":hashed_password", $hashed_password, PDO::PARAM_STR);
```

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

Les attaques XSS (Cross-Site Scripting) surviennent lorsque des pirates injectent du code malveillant dans un site web. Ce code est ensuite exécuté par le navigateur des utilisateurs qui visitent le site, entraînant des risques de sécurité.

```
<?php
// Récupérer et filtrer le nom de l'utilisateur
$nomUtilisateur = filter_input(INPUT_POST, 'nomUtilisateur', FILTER_SANITIZE_STRING);

// Afficher le nom de l'utilisateur de manière sécurisée
echo "<p>Nom : " . htmlspecialchars($nomUtilisateur) . "</p>";
?>
```

7.4. Protection contre les injections SQL

Les injections SQL se produisent lorsque des personnes malveillantes essaient d'insérer du code SQL non autorisé dans une requête de base de données. Cela peut entraîner des problèmes de sécurité importants.

```
// Prépare la requête SQL pour récupérer les informations de l'utilisateur avec le login spécifié.
$query = "SELECT id_utilisateur, nom_utilisateur, prenom_utilisateur, login, password FROM UTILISATEUR WHERE login = :login";
$stmt = $connexion->prepare($query); // Prépare la requête avec PDO.
$stmt->bindParam(":login", $login, PDO::PARAM_STR); // Lie la variable :login à la valeur du login, évitant les injections SQL.
```