

Cahier des charges techniques

JO 2024



PARIS 2024



Dossier rédiger par BAIDAH Amine

Sommaire

1.	Contexte du projet	3
1.1.	Présentation du projet	3
1.2.	Date de rendu du projet.....	3
2.	Besoins fonctionnels.....	3
3.	Ressources nécessaires à la réalisation du projet	3
3.1.	Ressources matérielles	3
3.2.	Ressources logicielles.....	3
4.	Gestion du projet	4
5.	Conception du projet	4
5.1.	Le front-end.....	4
5.1.1.	Wireframes	4
	Page accueil admin	7
	7
5.1.2.	Maquettes.....	8
	Page accueil admin	9
	9
	Page accueil admin.....	12
5.1.3.	Arborescences.....	13
5.2.	Le back-end	13
5.2.1.	Diagramme de cas d'utilisation	13
5.2.2.	Diagramme d'activités	14
5.2.3.	Modèles Conceptuel de Données (MCD).....	14
5.2.4.	Modèle Logique de Données (MLD)	15
5.2.5.	Modèle Physique de Données (MPD).....	16
6.	Technologies utilisées	17
6.1.	Langages de développement Web	17
6.2.	Base de données	17
7.	Sécurité	18
7.1.	Login et protection des pages administrateurs	18
7.2.	Cryptage des mots de passe avec Bcrypt	19
7.3.	Protection contre les attaques XSS (Cross-Site Scripting)	19
7.4.	Protection contre les injections SQL.....	19

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 sont : un PC portable et une connexion internet

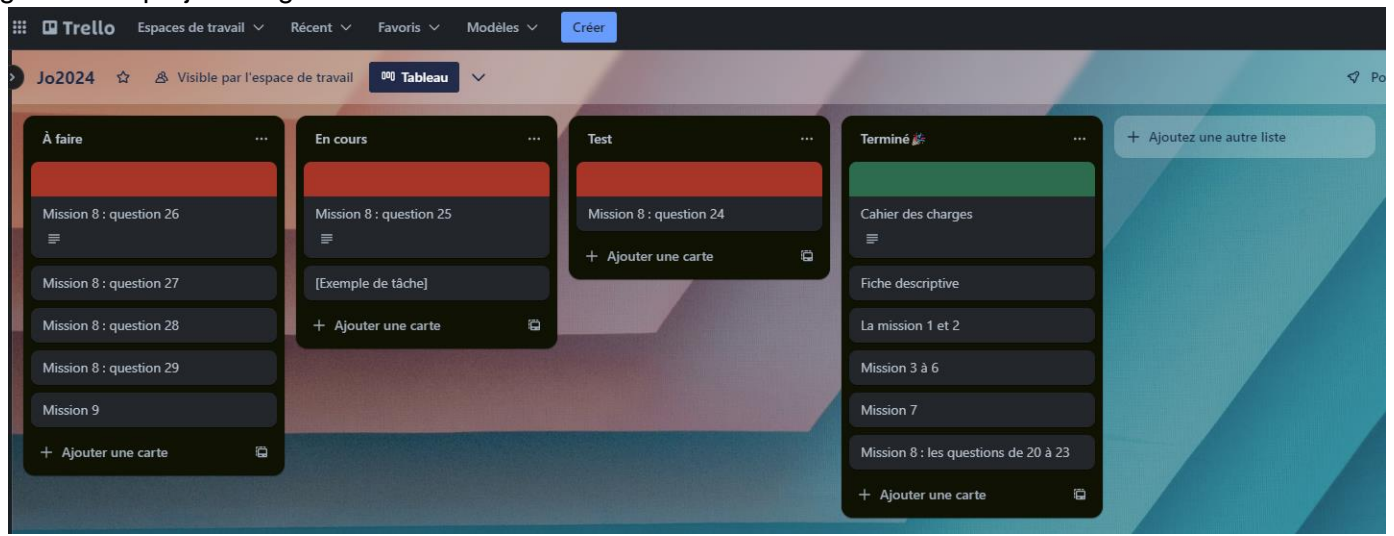
3.2. Ressources logicielles

Les ressources logicielles sont :



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.

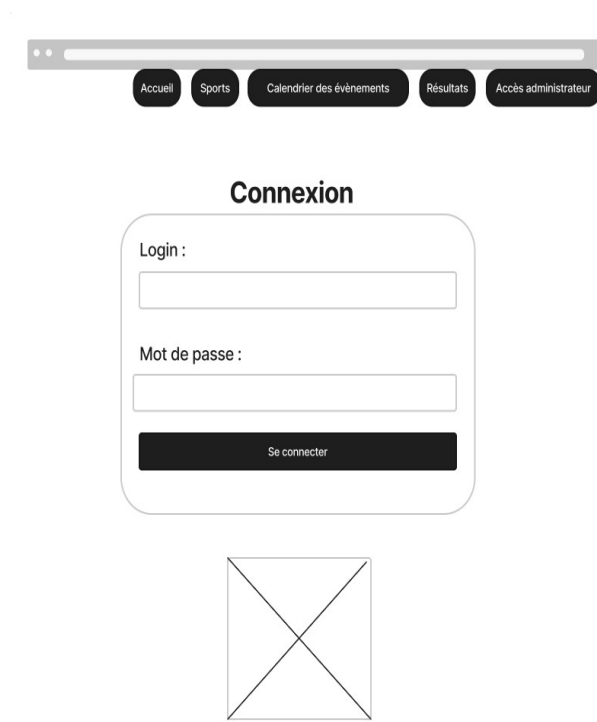
5. Conception du projet

5.1. Le front-end

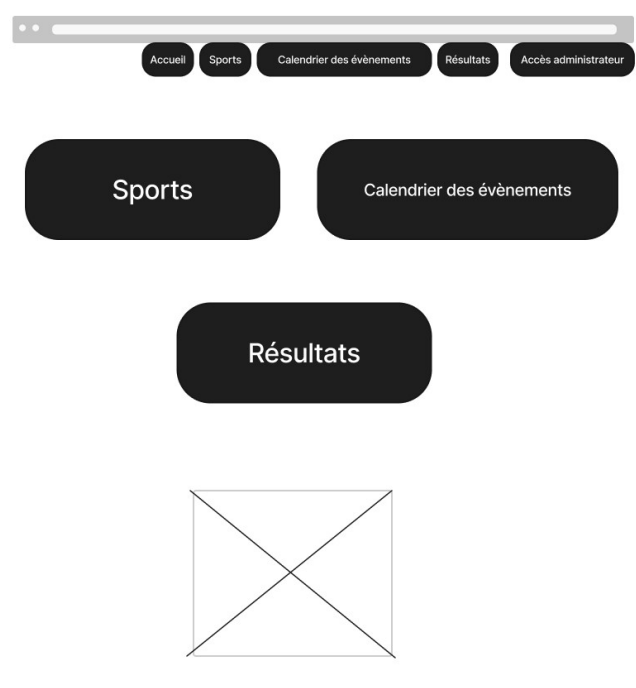
5.1.1. Wireframes

Mode PC :

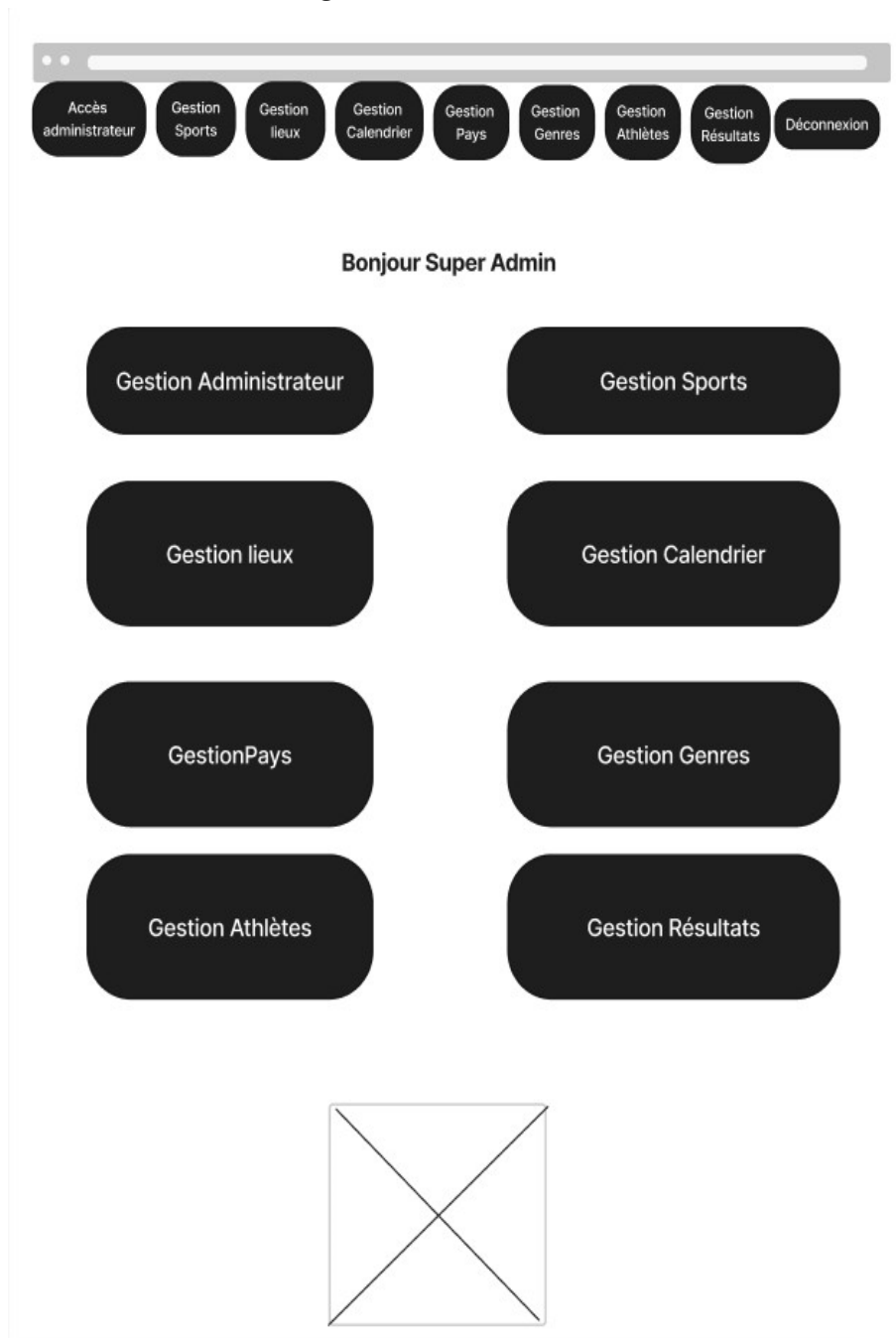
Page connexion



Page index

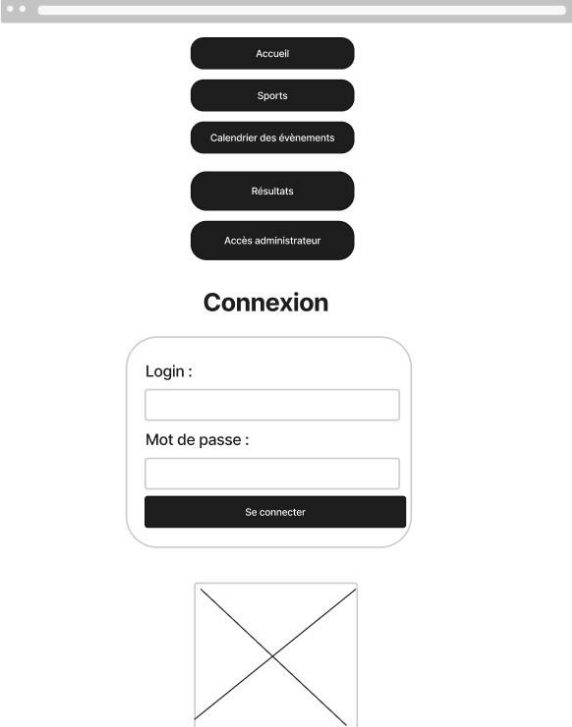


Page accueil admin



Mode responsive :

Page connexion



A wireframe of a responsive login page in mobile view. At the top, there is a browser window header with three dots and a progress bar. Below this is a vertical stack of five dark rounded rectangular buttons with white text: 'Accueil', 'Sports', 'Calendrier des événements', 'Résultats', and 'Accès administrateur'. Below the buttons is the heading 'Connexion'. Under the heading is a login form with a 'Login :' label and a text input field, followed by a 'Mot de passe :' label and another text input field. Below the inputs is a dark button with the text 'Se connecter'. At the bottom of the page is a large square placeholder with a diagonal 'X' inside.

Page index



A wireframe of a responsive index page in mobile view. It features a vertical stack of five dark rounded rectangular buttons with white text: 'ACCEUIL', 'SPORTS', 'CALENDRIER ET EVENEMENTS', 'RESULTATS', and 'ACCES ADMINISTRATEUR'. Below this stack are three more dark rounded rectangular buttons with white text: 'SPORTS', 'CALENDRIER DES EPREUVES', and 'RESULTATS'. At the bottom of the page is a large square placeholder with a diagonal 'X' inside.

Page accueil admin

Accès administrateur

Gestion Sports

Gestion lieux

Gestion Calendrier

Gestion Pays

Gestion Genres

Gestion Athlètes

Gestion Résultats

Déconnexion

Bonjour Super Admin

Gestion Administrateur

Gestion Sports

Gestion lieux

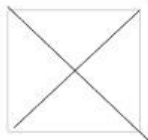
Gestion Calendrier

Gestion Pays

Gestion Genres

Gestion Athlètes

Gestion Résultats



5.1.2. Maquettes

Mode PC

Page connexion

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

Connexion

Login :

Mot de passe :

Se connecter



Page index

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

Page accueil admin

- Accueil Administration
- Gestion Sports
- Gestion Lieux
- Gestion Calendrier
- Gestion Pays
- Gestion Genres
- Gestion Athlètes
- Gestion Résultats
- Déconnexion

Bonjour Super Admin

Gestion Administrateurs

Gestion Sports

Gestion Lieux

Gestion Calendrier

Gestion Pays

Gestion Genres

Gestion Athlètes

Gestion Résultats



Mode responsive

Page connexion

Accueil

Sports

Calendrier des évènements

Résultats

Accès administrateur

Connexion

Login :

Mot de passe :

Se connecter



Page index

[Accueil](#)

[Sports](#)

[Calendrier des épreuves](#)

[Résultats](#)

[Accès administrateur](#)

[Sports](#)

[Calendrier des épreuves](#)

[Résultats](#)



Page accueil admin

Accueil Administration

Gestion Sports

Gestion Lieux

Gestion Calendrier

Gestion Pays

Gestion Genres

Gestion Athlètes

Gestion Résultats

Déconnexion

Bonjour Super Admin

Gestion Administrateurs

Gestion Sports

Gestion Lieux

Gestion Calendrier

Gestion Pays

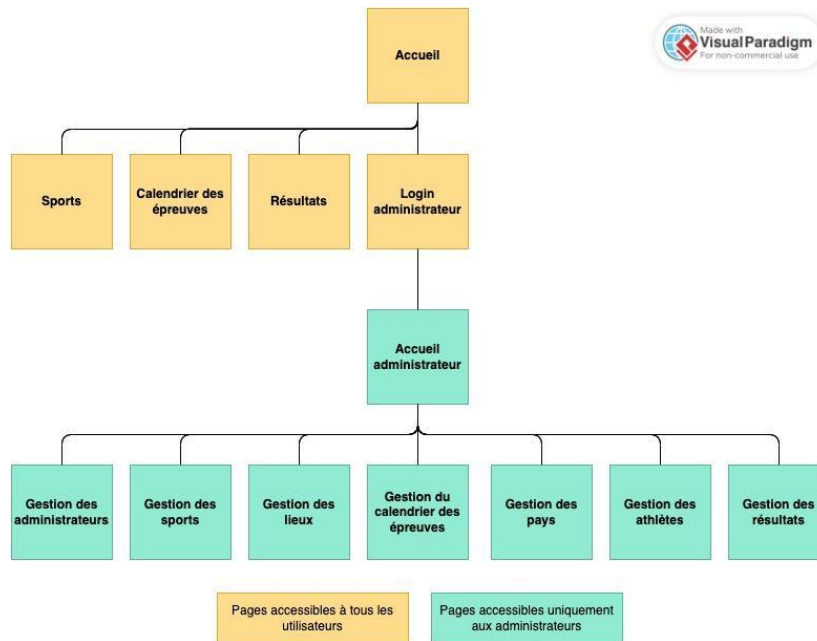
Gestion Genres

Gestion Athlètes

Gestion Résultats

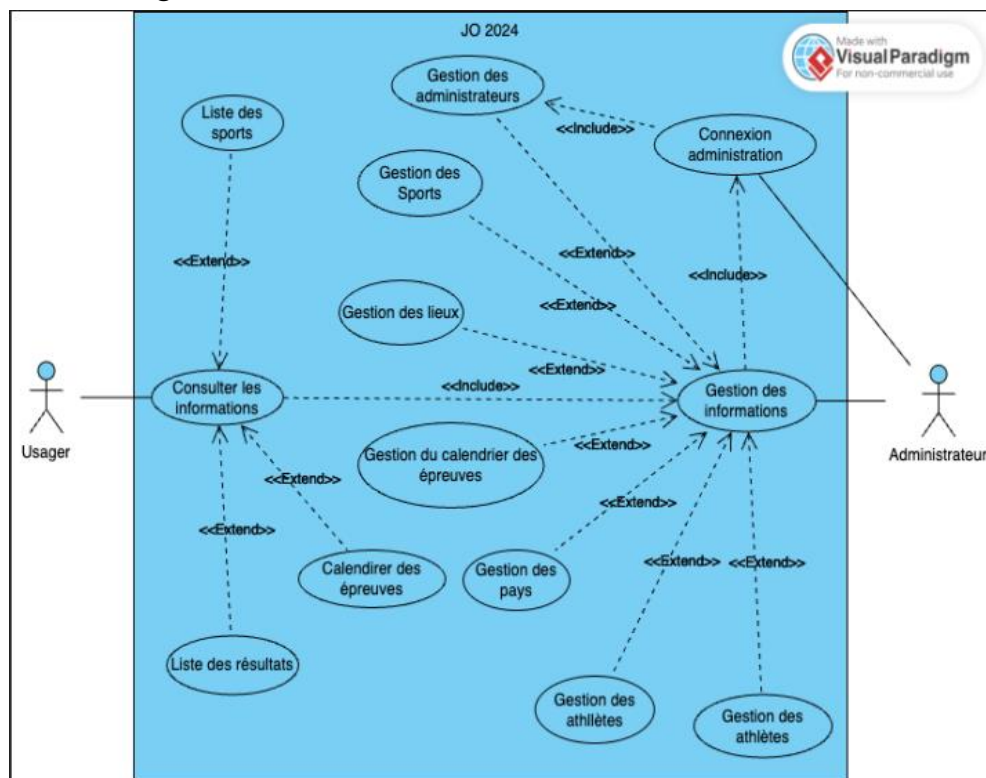


5.1.3. Arborescences

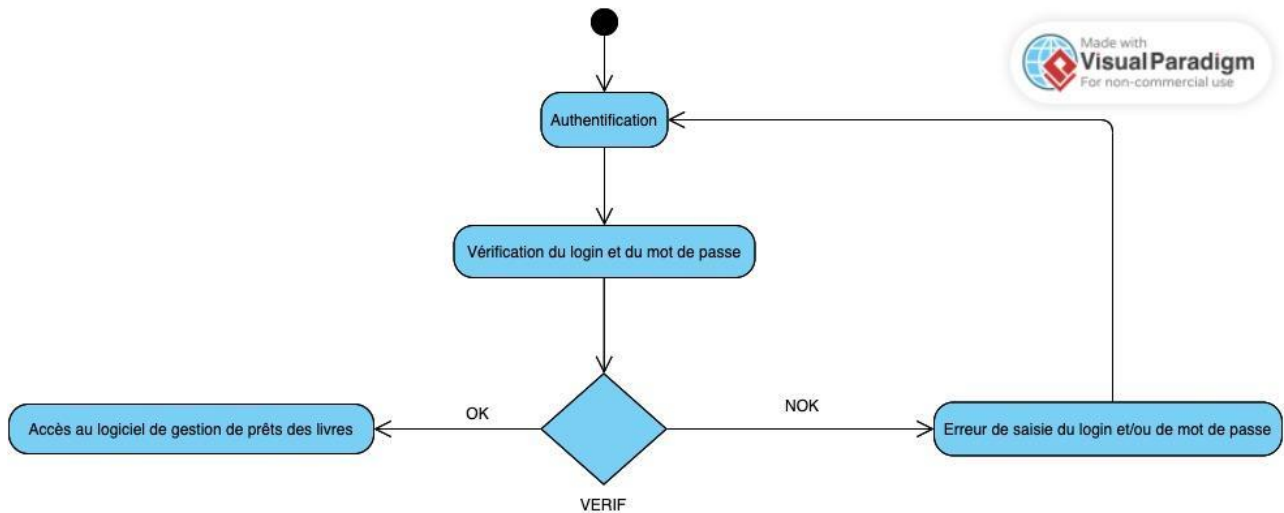


5.2. Le back-end

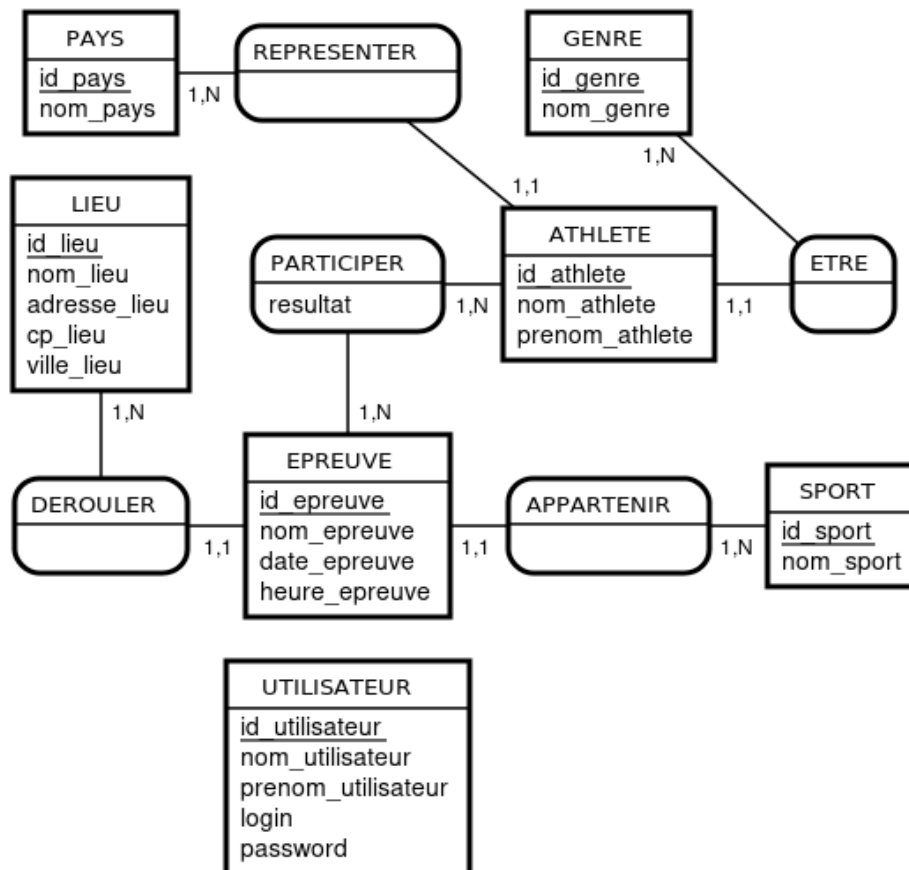
5.2.1. Diagramme de cas d'utilisation



5.2.2. Diagramme d'activités



5.2.3. Modèles Conceptuel de Données (MCD)



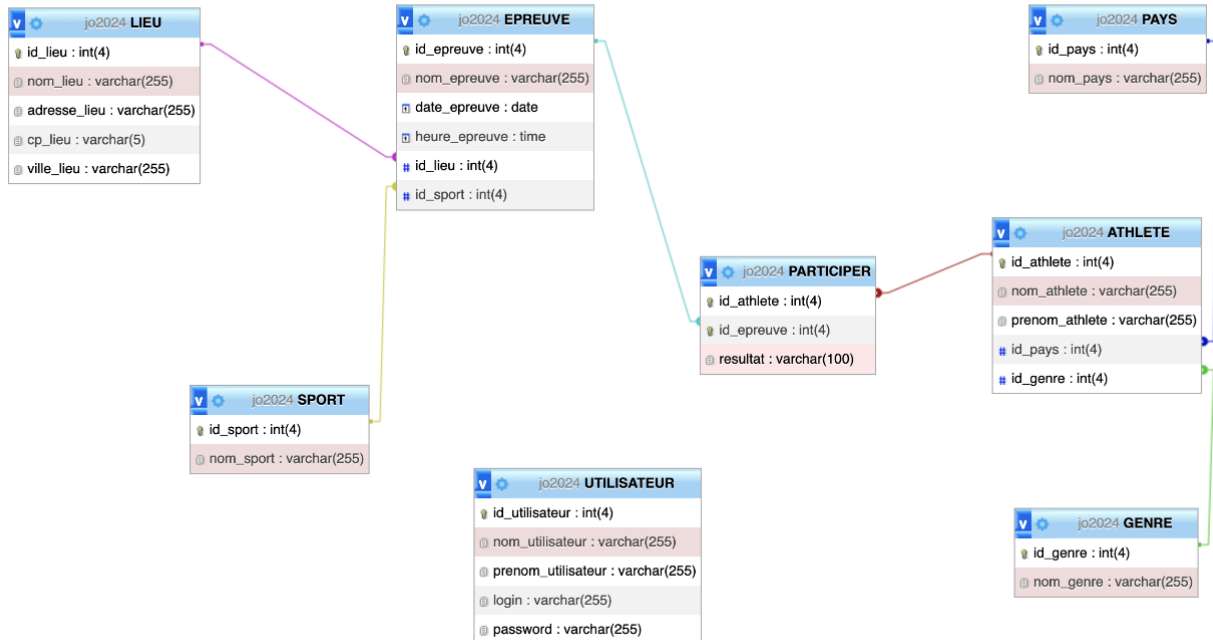
5.2.4. Dictionnaire des données

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)

5.2.5. 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)

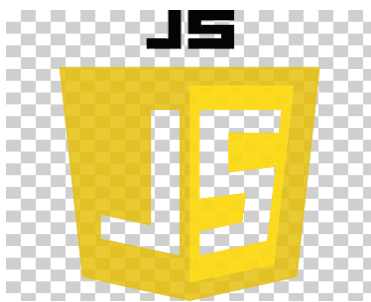
5.2.6. Modèle Physique de Données (MPD)



6. Technologies utilisées

6.1. Langages de développement Web

Les langages web utilisé sont :



6.2. Base de données



7. Sécurité

7.1. Login et protection des pages administrateurs

Tout d'abord on crée une page de connexion avec un formulaire (on utilise HTML et PHP) pour recueillir les identifiants . Si l'utilisateur n'a pas réussi à se connecter connecté, redirigez-le vers la page de connexion avec un message d'erreur . Voici le code de ma page login :

```
<main>
  <h1>Ajouter un Utilisateur</h1>
  <?php
  if (isset($_SESSION['error'])) {
    echo '<p style="color: red;">' . $_SESSION['error'] . '</p>';
    unset($_SESSION['error']);
  }
  ?>
  <form action="add-users.php" method="post"
    onsubmit="return confirm('Êtes-vous sûr de vouloir ajouter cet utilisateur ?')">
    <label for="nomUtilisateur">Nom de l'utilisateur :</label>
    <input type="text" name="nomUtilisateur" id="nomUtilisateur" required>
    <label for="prenomUtilisateur">Prénom de l'utilisateur :</label>
    <input type="text" name="prenomUtilisateur" id="prenomUtilisateur" required>
    <label for="login">Login :</label>
    <input type="text" name="login" id="login" required>
    <label for="mdp">Mot de passe :</label>
    <input type="password" name="mdp" id="mdp" required>
    <input type="submit" value="Ajouter l'utilisateur">
  </form>
  <p class="paragraph-link">
    <a class="link-home" href="manage-users.php">Retour à la gestion des Administrateurs</a>
  </p>
</main>
<footer>
  <figure>
    
  </figure>
</footer>
</body>
</html>
```

```
// Hachage du mot de passe avec Bcrypt
$hashed_password = password_hash($mdp, PASSWORD_BCRYPT);

// Requête pour ajouter un utilisateur
$query = "INSERT INTO UTILISATEUR (nom_utilisateur, prenom_utilisateur, login, password) VALUES (:nomUtilisateur, :prenomUtilisateur, :login, :hashed_password)";
$stmt = $connexion->prepare($query);
$stmt->bindParam(":nomUtilisateur", $nomUtilisateur, PDO::PARAM_STR);
$stmt->bindParam(":prenomUtilisateur", $prenomUtilisateur, PDO::PARAM_STR);
$stmt->bindParam(":login", $login, PDO::PARAM_STR);
$stmt->bindParam(":hashed_password", $hashed_password, PDO::PARAM_STR);

// Exécutez la requête
if ($stmt->execute()) {
  $_SESSION['success'] = "L'utilisateur a été ajouté avec succès.";
  header("Location: manage-users.php");
  exit();
} else {
  $_SESSION['error'] = "Erreur lors de l'ajout de l'utilisateur.";
  header("Location: add-users.php");
  exit();
}
```

7.2. Cryptage des mots de passe avec Bcrypt

Bcrypt est un algorithme qui permet le hachage de mots de passe sécurisés. En PHP, pour utiliser Bcrypt, utilisez la fonction `password_hash($mot_de_passe, PASSWORD_BCRYPT)` pour hacher le mot de passe, et `password_verify($mot_de_passe, $hash)` pour vérifier le mot de passe haché. Cela permet de renforcer la sécurité. Voici un extrait de mon code :

```
try {
    // Vérifiez si l'utilisateur existe déjà
    $queryCheck = "SELECT id_utilisateur FROM UTILISATEUR WHERE nom_utilisateur = :nomUtilisateur AND prenom_utilisateur = :prenomUtilisateur";
    $statementCheck = $connexion->prepare($queryCheck);
    $statementCheck->bindParam(":nomUtilisateur", $nomUtilisateur, PDO::PARAM_STR);
    $statementCheck->bindParam(":prenomUtilisateur", $prenomUtilisateur, PDO::PARAM_STR);
    $statementCheck->execute();

    if ($statementCheck->rowCount() > 0) {
        $_SESSION['error'] = "L'utilisateur existe déjà.";
        header("Location: add-users.php");
        exit();
    } else {
        // Hachage du mot de passe avec Bcrypt
        $hashed_password = password_hash($mdp, PASSWORD_BCRYPT);

        // Requête pour ajouter un utilisateur
        $query = "INSERT INTO UTILISATEUR (nom_utilisateur, prenom_utilisateur, login, password) VALUES (:nomUtilisateur, :prenomUtilisateur, :login, :hashed_password)";
        $statement = $connexion->prepare($query);
        $statement->bindParam(":nomUtilisateur", $nomUtilisateur, PDO::PARAM_STR);
        $statement->bindParam(":prenomUtilisateur", $prenomUtilisateur, PDO::PARAM_STR);
        $statement->bindParam(":login", $login, PDO::PARAM_STR);
        $statement->bindParam(":hashed_password", $hashed_password, PDO::PARAM_STR);

        // Exécutez la requête
        if ($statement->execute()) {
            $_SESSION['success'] = "L'utilisateur a été ajouté avec succès.";
            header("Location: manage-users.php");
            exit();
        } else {
            $_SESSION['error'] = "Erreur lors de l'ajout de l'utilisateur.";
            header("Location: add-users.php");
            exit();
        }
    }
} catch (PDOException $e) {
    $_SESSION['error'] = "Erreur de base de données : " . $e->getMessage();
    header("Location: add-users.php");
    exit();
}
```

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

L'attaque XSS (Cross-Site Scripting) a pour but d'injecter des scripts malveillants dans des pages web, exploitant des failles de sécurité. En PHP, pour se protéger, on utilise la fonction `htmlspecialchars($data, ENT_QUOTES, 'UTF-8')` pour échapper les caractères spéciaux dans les données utilisateur avant de les afficher, prévenant ainsi l'exécution de scripts indésirables.

```
<?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 ont pour but d'altérer ou extraire des données non autorisées dans une base de données. En PHP, on utilise les fonctions comme `PDO` et `mysqli` avec des placeholders pour filtrer et échapper automatiquement les données utilisateur, prévenant ainsi les injections SQL.

```
// 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.
```